User's Manual of **CAIN**

Version 2.1e Nov.1.1999

Contents

1 Introduction

CAIN is a stand-alone FORTRAN Monte-Carlo code for the interaction involving high energy electron, positron, and photons. Originally, it started with the name **ABEL**[1] in 1984 for the beam-beam interaction in e^+e^- linear colliders. At that time the main concern was the beam deformation due to the Coulomb field and the synchrotron radiation (beamstrahlung). Later, the pair creation by particle-particle collision was added, and, it was renamed to **CAIN** when the interaction with laser beams (radiation by electrons/positrons and pair creation by photons in a strong laser field) was added for the γ-γ colliders.

CAIN home page is located at http://www-acc-theory.kek.jp/members/cain/

The first version **CAIN**1.1[2], which is a combined program of modified **ABEL** and a laser QED code, is limited because it cannot handle the laser interaction and the e^+e^- interaction simultaneously and does not accept mixed e^+e^- beams. To overcome these problems, **CAIN**2.0 was written from scratch. It now allows any mixture of e^-, e^+, γ and lasers, and multiple-stage interactions. The input data format has been refreshed completely.

The physical objects which appear in the present version **CAIN**2.1e are two particle beams, lasers, and external fields. Each of the two beams may consist of high-energy electrons, positrons and photons. One of the beams may be absent. A basic assumption is that each beam must be a 'beam', *i.e.*, most particles in each beam go almost parallel. (**CAIN** assumes the two beams go opposite direction. For the case they make a large angle, you can apply **CAIN** command for Lorentz transformation so that the collision looks head-on.) The lasers can go any direction. The present version accepts only constant external fields.

The interactions that can be treated by the present version **CAIN**2.1e are

- Classical interaction (orbit deformation) due to the Coulomb field.
- Luminosity between (e^- e^+ γ).
- Synchrotron radiation (beamstrahlung), and pair creation by high energy photons (coherent pair creation) due to the beam field.
- Interaction of high energy photon or electron/positron beams with laser field, including the nonlinear effect of the field strength.
- Classical and quantum interactions with a constant external field.
- Incoherent e^+e^- pair creation by photons, electrons and positrons.
- In almost all interactions the polarization effects can be included.

1.1 General Structure of Input Data

In this section we briefly describe the structure of input data. **CAIN** is not intended for interactive jobs because the computing time is normally more than several minutes. Every intruction to the program is given in the input data. Two cases, a simple e^+e^- collision and a γ-γ collider, are given here as examples. For more detail look at the sections for each command and the example input data files in the directory `cain21e/in`.

Consider a simple e^+e^- collision. You have first to define the two beams:

```
 BEAM  RIGHT, KIND=2, NP=10000, AN=1E10, E0=500E9, SIGT=1E-4,
      BETA=(1E-2,1E-4), EMIT=(3E-12,3E-14);
```

This defines the right-going electron (`KIND=2`) with the bunch population 1×10^{10}, energy 500GeV, bunch length 100μm, etc. Note that every command must end with a semicolon.

You can use variables and mathematical expressions (see Sec.2.5). For example, if you prefer normalized emittance, you may write

```
 SET  ee=500E9,  gamma=ee/Emass, emitx=3D-6/gamma,  emity=3D-8/gamma,
      betax=1E-2, betay=1E-4,
      sigx=Sqrt(emitx*betax), sigy=Sqrt(emity*betay);
 BEAM  RIGHT, KIND=2, NP=10000, AN=1E10, E0=ee, SIGT=1E-4,
      BETA=(betax,betay), EMIT=(emitx,emity);
```

`Emass` is a reserved variable and `Sqrt` is a predefined function. `sigx` and `sigy` are defined for later use. If you like millimeter instead of meter, you may say

```
 SET    mm=1E-3, sigz=0.1*mm;
 BEAM   ........ SIGT=sigz, ......;
```

Now you know how to define the positron (`KIND=3`) beam. Obviously, `BEAM LEFT, KIND=3, ...;` will do.

For calculating the beam-beam force you need to tell **CAIN** about the mesh:

```
SET Smesh=sigz/2;
BBFIELD  NX=32, NY=32, WX=8*sigx, R=sigx/sigy/2;
```

The definition of the longitudinal mesh `Smesh` may look bizzarre. This is because the same mesh is used for luminosity calculation.

For computing the e^+e^- luminosity, you have to say, for example,

```
 LUMINOSITY KIND=(2,3), W=(0,2*ee,50), WX=8*sigx, WY=8*sigy, FREP=90*150;
```

if the rep rate is 90 bunches times 150Hz. `WX` and `WY` define the mesh region (See Sec.3.9).

Now you are ready to start the collision.

```
 FLAG OFF ECHO;
 PUSH  Time=(-2.5*sigz,2.5*sigz,200);
 ENDPUSH;
```

will track the beam over the specified time range in 200 steps. It is better to turn off the echo before running. You can get the transient information (e.g., plot the beam profile during collision) by inserting commands (`PLOT`, `WRITE` etc) between `PUSH` and `ENDPUSH`. If you want the bamstrahlung, you have to insert

```
 CFQED    BEAMSTRAHLUNG;
```

before `PUSH`. After `ENDPUSH` you can plot (generate TopDrawer input file) the e^+e^- differential luminosity by

```
 PLOT LUMINOSITY, KIND=(2,3);
```

You can also plot particle distribution. For example, for plotting the photon (`KIND=1`) energy spectrum,

```
 PLOT  HIST, KIND=1, H=En/1E9, HSCALE=(0,ee/1E9,50),
        TITLE='Beamstrahlung Energy Spectrum;',
        HTITLE='EOG1  (GeV); XGX       ;';
```

`H` defines the horizontal axis (energy in units of GeV).

You may want various outputs without repeating the time-consuming calculation. You can do the following. After `ENDPUSH`, store all the variables and the particle data:

```
STORE FILE='aaa';
WRITE BEAM, FILE='bbb';
```

and restore in another input file

```
RESTORE FILE='aaa';
BEAM  FILE='bbb';
PLOT  ........;
```

γ-γ collider is more complex. Three steps, e-γ conversion of right-going electron, that of left-going electron, and γ-γ collision, are needed. You can do these steps in one job or in separate jobs using `STORE/WRITE` and `RESTORE/BEAM FILE` commands. The attached example `cain21e/in/NLCggCP.i` executes the two conversions and `NLCggIP.i` the collision at the interaction point.

For the conversion you define the lasers in addition to the initial electron beam:

```
 LASER LEFT, WAVEL=laserwl, POWERD=powerd,
      TXYS=(-dcp,0,off/2,-dcp),
      E3=(0,-Sin(angle),-Cos(angle)), E1=(1,0,0),
      RAYLEIGH=(rlx,rly), SIGT=sigt, STOKES=(0,1,0) ;
```

See Sec.3.4 for the meaning of the key words. The type of laser-electron and laser-γ interactions has to be specified by `LASERQED` command:

```
 LASERQED  COMPTON, NPH=5, XIMAX=1.1*xi, LAMBDAMAX=1.1*lambda ;
 LASERQED  BREITW, NPH=5, XIMAX=1.1*xi, ETAMAX=1.1*eta ;
```

The `PUSH-ENDPUSH` loop is the same as in the e^+e^- example.

After `ENDPUSH` write all the particle data by `WRITE BEAM, FILE=...` or, if you do not want to include e-e collision, write the photon data selectively by `WRITE BEAM, KIND=1, FILE=....` Then, read this file in the next job and simulate the γ-γ collision.

See Sec.2 for the basic grammer of the input data. See Sec.2.4 for a list of all the available commands.

1.2 Change since the last version CAIN2.1b

There has been a version **CAIN**2.1d but the changes since then are only bug fixes. Here we list up the changes since **CAIN**2.1b.

- Unequal mesh of energy for differential luminosity has been introduced.

1.3 History until the last version CAIN2.1b

New entries on physics

- 2D differential luminosity $d\mathcal{L}/dE_1dE_2$ added.
- Lorentz transformation of lasers has been added.
- Field-strength dependence of the anomalous magnetic moment of electron is taken into account in solving the Thomas-BMT equation.
- Polarization dependence of the beamstrahlung and the coherent pair creation has been included.
- The kinematics in nonlinear QED subroutines was improved so as to accept non-relativistic electrons/positrons.
- The final polarization of electron in the nonlinear Compton scattering was added.
- Polarization change in linear and nonlinear QED, beamstrahlung and coherent pair creation processes when event generation is rejected is now taken into account.
- Incoherent $\mathrm{e}^+\mathrm{e}^-$ pair creation by Breit-Wheeler, Bethe-Heitler, and Landau-Lifshitz processes has been added.
- Luminosity with full polarization information (including linear polarization) can be computed.
- Differential luminosities with unequal-space energy bins are introduced.

New entries on user interface

- Following pre-defined variables have been added:
  `Kind`, `Gen`
- Following pre-defined functions have been added:
  ```
   Min, Max,
   AvrT, AvrX, AvrY, AvrS, SigT, SigX, SigY, SigS,
   AvrEn, AvrPx, AvrPy, AvrPs, SigEn, SigPx, SigPy, SigPs,
   TestT, TestX, TestY, TestS, TestEn, TestPx, TestPy, TestPs,
   LumP, LumWP, LumWbin, LumWbinEdge,
  LumEE, LumEEbin, LumEEbinEdge, LumEEH, LumEEP,
   BesJ, DBesJ, BesK13, BesK23, BesKi13, BesKi53,
   FuncBS, FuncCP, IntFCP
  ```

- Do-type sequence in PRINT/WRITE command became possible.
- Maximum number of characters of user parameters is increased to 16. Also, the underscore '_' is allowed in parameter names.
- The flags for beamstrahlung and coherent pair creation, which had been defined in the BBFIELD command, were moved to a new command CFQED (constant-field QED). This is more logical becuase **CAIN** computes these phenomenon due to the external fields, too.[1] Acoording to this change, CFQED operand was added to CLEAR command. Except for this change, input data files prepared for **CAIN**2.1b can be used for **CAIN**2.1e.
- New commands STORE and RESTORE were added. You can store the variables and the luminosity data for later jobs.
- Command PLOT FUNCTION was added.

Bugs (already fixed in the present version)

- There was a bug in CLEAR BEAM command when applied during a PUSH ENDPUSH loop. Fixed.
- A bug was found in the file source/physics/bb/bbmain/bbkick.f in solving the equation of motion under the beam-beam force. It is a kind of double counting of the beam-beam effect. Fixed.
- Several bugs were found in DRIFT, EXTERNAL command. Fixed.
- There was a miss-spelled variable in subroutine EVUFN (in the directory source/control/deciph). (This has been overlooked because of missing IMPLICIT NONE.) Not very harmful. Fixed.
- Total helicity luminosity was not calculated, although differential helicity luminosity was. (A bug in physics/lum/dlumcal.f) Fixed.
- PRINT/WRITE command did not correctly understand the KIND operand. Fixed.
- The polarization sign of the final positron in the subroutine for linear Breit-Wheeler process (source/physics/laser/nllsr/lnbwgn.f) was wrong. Corrected.
- Linear polarization of final photons in the linear Compton scattering was wrong. (ξ_1, ξ_3) used to come out as $(-\xi_3, \xi_1)$. Fixed.

2 Basic Grammer of the Input Data

2.1 System of Units

MKSA is used throughout. The particle energy and momentum are eV and eV/c, respectively. An exception is the luminosity which is expressed in cm^{-2}sec^{-1}. The time (e.g., the laser pulse length, time coordinate of particles, etc.) is always expressed in units of meter by multiplying the velocity of light.

[1] Following this logic more faithfully, **CAIN** should have adopted the word SYNCHROTRONRADIATION instead of BEAMSTRAHLUNG.

2.2 Characters

Upper and lower case alphabets are distinguished. The following characters have special use:

`= ; , ( [ { ) ] } ! < > '`

Also, the following characters are used in mathematical expressions:

`+ - * / ^ = . ( [ { ) ] }`

2.3 File Lines and Command Blocks

The input data is a collection of file lines. Upto 256 characters in a line are read in. (This limitation can be easily changed by modifying the parameter statement in the main program.)

If a character "!" is encountered, the whole text after it to the end of the file line is considered as a comment, unless the "!" is in a pair of apostrophes. Apostrophe pairs must close within a file line except for comment parts. Apart from the above two points, the concept of 'file line' is irrelevant. Therefore, for example, continuing the two file lines will give the same results, and the end of a command must explicitly said (by semicolon ";") without relying on the end-of-line.

The whole text, after comment part is eliminated, is divided into 'command blocks'. The end of a command block is indicated by a semicolon ";". If ";" is inside a pair of apostrophes, it is not considered to be a command terminator.

Each command block has the following structure:

command_name operand, operand, ⋯ operand ;

After the command_name before the first operand, there must be at least one blanck character (unless there is no operand). Operands are separated by a comma "," and the number of blancks before and after "," is arbitrary. (In some commands, "," can be replaced by one or more blancks). Unless stated in each command description in the next section, the order of operands is arbitrary.

An operand is either a single keyword or of the form

keyword relational_operator right_hand_side

A keyword is an alphanumerical string predefined for each command. The relational operator is either one of =, <, >, <=, >=, =<, =>, <>, ><. The right_hand_side is just a number or an 'expression' (to be explained later) or of the form

(expression, expression, ⋯ expression)

The parenthesis ( ) may be replaced by [ ] or { }.

2.4 Commands

As stated above, each command block must start with a command name. The present version has the following commands

FLAG	On-off flags (echo, etc.). Sec.3.1.
SET	Define user variables. Sec.3.2.
BEAM	Define particle beams. Sec.3.3.

`LASER`	Define lasers. Sec.3.4.
`EXTERNALFIELD`	Define external (static) electromagnetic field. Sec.3.8.
`LASERQED`	Parameters for the laser-particle interaction. Sec.3.5.
`CFQED`	Parameters for the interaction between particles and constant electromagnetic field (beamstrahlung and coherent pair creation). Sec.3.6.
`BBFIELD`	Method of calculation of the beam field. Sec.3.7.
`PPINT`	Incoherent particle-particle interaction. Sec.3.10.
`LUMINOSITY`	Define what sort of luminosities to be calculated. Sec.3.9.
`LORENTZ`	Lorentz transformation. Sec.3.13.
`DRIFT`	Move particles in vaccuum or in external field. Sec.3.12.
`PUSH`,`ENDPUSH`	Loop of time steps. Sec.3.11.
`DO`,`ENDDO`	Do loop. Sec.3.14.
`IF`,`ELSE`,`ENDIF`	If block. Sec.3.15.
`WRITE`,`PRINT`	Print on screen or on a file. Sec.3.16.
`PLOT`	Plot using TopDrawer. Sec.3.17.
`CLEAR`	Clear data or disable commands. Sec.3.18.
`FILE`	Open/close files. Sec.3.19.
`HEADER`	Define the header for graphic outputs. Sec.3.20.
`STORE`,`RESTORE`	Save/recall variables and luminosity values. Sec.3.21.
`STOP`	Stop run. Sec.3.22.
`END`	End of the input file. Sec.3.23.

The command names may be shortened if not ambiguous. Therefore, `LASERQ` is equivalent to `LASERQED`. This rule applies also to the operand keywords of all commands. (But does not apply to parameter and function names.)

2.5 Expressions

In the example in Sec.1.1, the right_hand_sides of some operands are written in the form of mathematical expressions. In general, 'expression' is a mathematical expression which is almost identical to FORTRAN floating expression. It may contain

- Literal numbers, such as `2`, `2.0`, `-3E-5`, etc.
  To indicate the exponent, any of `E,e,D,d,Q,q` may be used. Note that there is no integer expression so that `2` is identical to `2.0`.
- Operators `+`,`-`,`*`,`/`,`^`. Note that power is indicated by "`^`" instead of "`**`".
- Parenthesis: `( [ { ) ] }` . Must match.
- Pre-defined parameters: There are three types of predefined parameters. The first type is the universal constants that never change:

Pi	π
E	$e = 2.718\ldots$
Euler	Euler's constant $\gamma_E = 0.577\ldots$
Deg	$0.0174\ldots = \pi/180$. You can write, e.g., `10*Deg` where the randian unit is required.
Cvel	Velocity of light (m/sec).
Hbar	Planck's constant (Joule·sec).
Hbarc	Planck's constant times the velocity of light (eV·m).
Emass	Electron mass (eV/c^2).
Echarge	Elementary charge (Coulomb).
Reclass	Classical electron radius (m).
LambdaC	Compton wavelength (m).
FinStrC	Fine structure constant.

The second type is the parameters whose values are determined by the program. Users cannot change their values but can refer to.

T,X,Y,S	Running variables for particle coordinate (m).
En,Px,Py,Ps	Running variables for energy-momentum (eV, eV/c). The energy is `En` but not `E`.
Sx,Sy,Ss	Electron/positron spin. Helicity may be written approximately as `Ss*Sgn(Ps)`.
Xi1,Xi2,Xi3	Photon Stokes parameters ξ_1, ξ_2, ξ_3.
Kind	Particle species. 1,2,3 for photon, electron, positron.
Gen	Particle generation.
Time	Running variables for global time coordinate (m).
Ln,Lij	(n=0,1,2,3,4, i,j=0,1,2,3) Luminosity values used in `PLOT LUMINOSITY` command.
W	Center-of-mass energy used in `PLOT LUMINOSITY` command.

The third type is those whose names are predefined with default values and which the user can change (by `SET` command) such as

MsgFile	File reference number for echo, error messages, and default destination of `PRINT` command. (default=6)[2]
Outfile	File reference number for voluminous outputs. The default destination of `WRITE` command. (default=12)
OutFile2	Other print output. Not used. (default=12)
TDFile	TopDrawer file number. (default=8)
MsgLevel	Message level. (default=0, i.e., error messages only)

[2] The input file number is set to 5. If you want to change it, see the variable `RDFL` in the file 'cain21e/source/control/main/initlz.f'.

`Rand` Random number seed. Positive odd integer other than 1, default=12345. You can reset random number at any time.

`Debug` Debug parameter for the programmer. If you set `Debug`≥2, call and return from major subroutines are announced. (default=0)

`Smesh` Longitudinal mesh size (m) for the calculation of beam-beam field, luminosity, etc. No default value.

- User-defined parameters: Those defined by `SET` command. Upto 16 characters consisting of upper/lower case alphabets, numericals, and underscore '_'. The first character must not be a numerical.

- Predefined functions such as
  `Int,Nint,Sgn,Step,Abs,Frac,Sqrt,Exp,Log,Log10,`
  `Cos,Sin,Tan, ArcSin,ArcCos,ArcTan,`
  `Cosh,Sinh,Tanh, ArcSinh,ArcCosh,ArcTanh, Gamma,`
  `Mod, Atan2, Min, Max`
  Defintions are the same as in standard FORTRAN except `Sgn` and `Step`:
  `Sgn(`x`)`=1, 0, −1, for $x > 0$, $x = 0$, $x < 0$.
  `Step(`x`)`=1, 0, for $x \geq 0$, $x < 0$.
  Enclose the argument by `( )` or `[ ]` or `{ }`. Separate arguments by "`,`" if there are more than one argument (`Mod`, `Atan2`, `Min`, `Max`). (Number of arguments for `Min` and `Max` is arbitrary.)
  There are functions of other type, which are defined for **CAIN**. See the next subsection Sec.2.6.

2.6 CAIN functions

In addition to the predefined functions of general use, such as `Sin` and `Cos`, there are other special functions intrinsic to **CAIN**. There is one limitation on the arguments of the **CAIN** functions: the arguments must not contain any **CAIN** functions (due to the problem of FORTRAN recursive call).

Beam statistics functions

Firstly, the number of particles, that of macro particles, the average coordinates/energy-momentum and their r.m.s. values of the beam at the given moment are retrieved by
`NParticle, NMacro,`
`AvrT, AvrX, AvrY, AvrS, SigT, SigX, SigY, SigS,`
`AvrEn, AvrPx, AvrPy, AvrPs, SigEn, SigPx, SigPy, SigPs,`
`BeamMatrix` The calling sequence is common to these functions except `BeamMatrix`. Let us take `SigX` as an example.
`SigX(`j`,`k`)` (j= 1 or 2 or 3, k= 1 or 2 or 3)
returns the horizontal r.m.s. size of right-going (j=1) or left-going (j=2) or both (j=3) of the photon (k=1) or electron (k=2) or positron (k=3) beam. If there is one more argument like
`SigX(`j`,`k`,`s_0`)`

then the particles are restricted to those in the region $s_0 - \texttt{Smesh}/2 \le s \le s_0 + \texttt{Smesh}/2$. The variable Smesh must be defined before.

BeamMatrix requires two more arguments

BeamMatrix(a,b,j,k,s_0) ($1 \le a,b \le 8$)

The returned value is the average of $x_a x_b$ where x_a=(T,X,Y,S,En,Px,Py,Ps) for a=1 to 8. (In units of m and eV or eV/c.)

Test particle variables

The coordinates and the energy momentum of the test particles can be retrieved by the functions

TestT, TestX, TestY, TestS, TestEn, TestPx, TestPy, TestPs

The calling sequence is, for example, TestX('name') or TestX(n), where 'name' is the character string for the particle name and n is an expression representing an integer $-99 \le n \le 999$. (See Sec.3.3 for the test particle name.)

Luminosity-related function

There are functions related to the luminosity:

Lum, LumH, LumP

LumW, LumWbin, LumWbinEdge, LumWH, LumWP

LumEE, LumEEbin, LumEEbinEdge, LumEEH, LumEEP

See Sec.3.9 for definitions for these functions.

Special functions

Bessel function J_n. (n must be an integer.)

BesJ(n,x)	Bessel function $J_n(x)$. n must be an integer.
DBesJ(n,x)	Derivative of Bessel function, $J_n'(x)$.

Modified Bessel function K_ν and its integral. In all the following functions, the last argument k must be 1 or 2. When $k = 2$, the output is the function multiplied by e^x. The last argument may be omitted (equivalent to $k = 1$.)

BesK(ν,x,k)	Modified Bessel function $K_\nu(x)$. ($x > 0$)
DBesK(ν,x,k)	Derivative of the modified Bessel function $K_\nu'(x)$. ($x > 0$)
BesK13(x,k)	Modified Bessel function $K_{1/3}(x)$. ($x > 0$)
BesK23(x,k)	Modified Bessel function $K_{2/3}(x)$. ($x > 0$)
BesKi13(x,k)	Integral of Modified Bessel function, $Ki_{1/3}$. ($x > 0$) See eq.(94) for the definition of Ki.
BesKi53(x,k)	Integral of Modified Bessel function, $Ki_{5/3}$. ($x > 0$)

Functions for beamstrahlung and coherent pair creation.

`FuncBS(`x`,`Υ`)` Beamstrahlung function F_{00} defined in eq.(92). x $(0 < x < 1)$ is the photon energy in units of the initial electron energy. $\Upsilon > 0$.

`FuncCP(`x`,`χ`)` Spectrum function F_{CP} of coherent pair creation defined in eq.(108). x $(0 < x < 1)$ is the positron energy in units of the initial photon energy. $\chi > 0$.

`IntFCP(`χ`,`k`)` Integral of `FuncCP(`x`,`χ`)` over $0 < x < 1$. The total rate of coherent pair creation is given by multiplying by $\alpha m^2/(\sqrt{3}\pi E_\gamma)$. (See Sec.5.9). k must be 1 or 2 (can be omitted if 1). If $k = 2$, the function is multiplied by $\exp(8/3/\chi)$.

3 Commands

A command, in general, has the following structure:

command_name   op$_1$, op$_2$, ... , op$_n$;

A command_name is a string consisting of upper-case roman letters and numbers only. There must be one or more than one blanck characters after a command_name before the first operand.

'op$_j$' is an operand having either one of the following forms:

(a) kwd
(b) expr
(c) kwd rel expr
(d) expr rel expr

Here, 'rel' is a relational operator which is either one of `=`, `<`, `>`, `<=`, `>=`, `=<`, `=>`, `<>`, `><`. 'kwd' is a keywaod, i.e., a string consisting of upper-case roman letters only, which is predefined for each command. 'expr' is a mathematical expression described in Sec.2.5.

An operand of the form (a) is a flag-type operand. In some commands, the first operand must be a positional operand of the flag-type. (For example, `DO` command must be either `DO WHILE` or `DO REPEAT`.) In such a case, the "`,`" after the keyword may be omitted. (There is no ambiguity because keywords do not contain blanck characters in contrast to expressions.) `FLAG` command is special in that all the commas may be omitted because all the operands are type(a).

The right-hand-side of type(c) can be `(`expr`,`expr`,`...`)`, or can be a character string for some operands

Now, let us describe the each command in detail. When describing the command formats in this manual, the type-faced characters are those to be typed in the input data as it is. (The variable names in the FORTRAN source also appear in type-face.) The items embraced by square brackets [ ] may be omitted in some cases and the vertical stroke "|" indicates an exclusive choice of one of the items. Thus, [`A`|`B`] means to choose either one of `A` or `B` or to omit both. Note that [ ] and `[ ]` are different.

The dagger † indicates that the operands to the left of it are positional operands.

The quantities printed in math-font in command syntax can be expressions.

3.1 FLAG

Set flag. example:

```
FLAG  ON ECHO OFF SPIN ;
```

The keywords `ON` and `OFF` act until the opposite one appears. `ON` is the default after `FLAG`.

Existing flags

`ECHO` input data echo (default=`ON`)

`SPIN` include spin calculation (default=`ON`) (Sorry, spin calculation cannot be avoided in the present version.)

3.2 SET

Defines parameters.

Syntax:

`SET` [p = *a*] [, p = *a*] ⋯ ;

p New or old parameter name. Upto 16 characters consisting of upper/lower case alphabets, numericals, and underscore '`_`'. The first character must not be a numerical. Unchangeable predefined parameters (`Pi`, `Time`, etc) and the predefined function names (`Sin`, etc) have to be avoided. All the predefined parameter names and function names start with an uppercase letter. Therefore, a user parameter starting with a lower case alphabet will never hit the predefined ones.

a An expression. See Sec.2.5.

3.3 BEAM

Defines a beam. (Append particles to the existing list.) There are two ways to create a beam, one by specifying the Twiss parameters, etc, and the other by reading data from a file.

Definition by Twiss parameters

Note that the beam is defined on a plane s=constant, rather than on the t=constant plane (snap shot). Thus, e.g., the bunch length is a spread in t rather than in s.

Syntax:

`BEAM` `RIGHT`|`LEFT,` `KIND=`k`,` `AN=`N`,` `NP=`N_p`,` `E0=`E_0`,`
[`TXYS=(`t`,`x`,`y`,`s`),`] `BETA=(`β_x`,`β_y`),` [`ALPHA=(`α_x`,`α_y`),`] [`EMIT=(`ϵ_x`,`ϵ_y`),`]
[`SIGT=`σ_t`,`] [`SIGE=`σ_ε`,`] [`GCUT=(`n_x`,`n_y`),`] [`GCUTT=`n_t`,`] [`GCUTE=`n_ε`,`]
[`GAUSSWEIGHT=`i_g`,`] [`ELLIPTIC,`] [`TUNIFORM,`] [`EUNIFORM,`]
[`SLOPE=(`θ_x`,`θ_y`),`] [`CRAB=(`ψ_x`,`ψ_y`),`] [`ETA=(`η_x`,`η_y`),`]
[`ETAPRIME=(`η'_x`,`η'_y`),`] [`ESLOPE=`$d\varepsilon/dt$`,`] [`XYROLL=`ϕ_{xy}`,`]
[`SPIN=(`ζ_x`,`ζ_y`,`ζ_s`),`] ;

`RIGHT|LEFT` Specify whether the beam is right-going or left-going.

k Particle species. 1 for photon, 2 for electron, 3 for positron. If you cannot remember these codes, you can do

```
SET photon=1, electron=2, positron=3 ;
BEAM RIGHT, KIND=electron  ...
```

N Number of real particles.

N_p Number of macro-particles.

E_0 Beam energy. (eV)

t,x,y,s Location of the reference point and the time when the beam center comes there. In units of meter. This is the point where the Twiss parameters are to be defined. Default=(0,0,0,0).

β_x,β_y Beta functions (m).

α_x,α_y Alpha functions. Default=(0,0). The sign of α is positive when the beam is going to diverge, whichever the beam is right-going or left-going.

ϵ_x,ϵ_y R.m.s geometric emittance (rad·m). Deafault=(0,0).

σ_t R.m.s. bunch length (m). Default=0.

σ_ε Relative r.m.s. energy spread. Default=0.

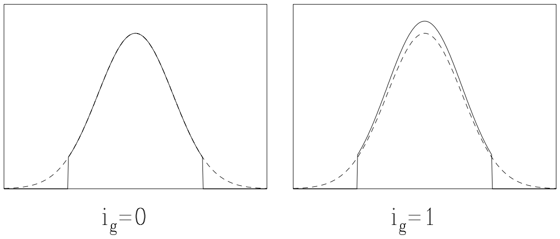
n_x,n_y,n_t,n_ε Gaussian tail cutoff in units of corresponding sigmas. The default values are 3.0 for n_x and n_y, and 2.5 for n_t and n_ε. (For transverse variables the cut off is done in the action variable, which means $J_i/\epsilon_i \leq n_i^2/2$ (i=x,y).)

i_g 0 or 1. There is a subtle problem on how to take into account the Gaussian cut off (n_x,n_y,n_t) in the macro-particle weight. **CAIN** throws away the random numbers outside this range and generates exactly N_p macro-particles. This means some fraction outside the region is moved inside. Therefore, if the simple weight N/N_p is assigned to macroparticles ($i_g = 1$), the effective particle density would become slightly larger than the physical value, although the sum of the weight is equal to N. If one is interested in the quantities related to the density (such as luminosities), this would cause an overestimation.

When $i_g = 0$ (default), a correction factor is multiplied to the weight such that the real particle density becomes correct. In this case, the sum of the macro-particle weights is less than N. (When the default n's are adopted, for example, the correction of the weight amounts to ~3.4%.) In most cases, $i_g = 0$ will be better.

`ELLIPTIC` Uniform transverse distribution. (Default is Gaussian.) (x, y) distribution is a uniform ellips with radii $(2\sigma_x, 2\sigma_y)$, where $\sigma_j = \sqrt{\epsilon_j \beta_j}$ (j=x,y). In this case the beam is parallel, in spite the finite emittances are specified. The emittance and beta are only used to define $\sigma_{x,y}$. `ALPHA` and `GCUT` are not used.

Figure 1: Physical charge density (dashed curve) and the simulated density (solid) for i_g=0 and 1



TUNIFORM Uniform t-distribution. (Default is Gaussian.) The full length is $2\sqrt{3}\sigma_t$. GCUTT is not used.

EUNIFORM Uniform E-distribution. (Default is Gaussian.) The full relative energy spread is $2\sqrt{3}\sigma_\varepsilon$. GCUTE is not used.

θ_x,θ_y Angle offset (radian). The right and left-going beams have the same sign of slope when there is a crossing angle. Default=(0,0).

ψ_x,ψ_y Crab angle $\partial x(y)/\partial t$. (radian). Positive when the bunch tail has larger x (y).

When the full crossing angle in the horizontal plane is ϕ_{cross} and this is to be compensated by the crab angle, the SLOPE and CRAB parameters should be SLOPE=$\phi_{cross}/2$, CRAB=$\phi_{cross}/2$, for both right-going and left-going beams.

If you are not confident, after beam definition try, for example,

```
 DRIFT T=t0-dt ;
 PLOT SCAT, H=S, V=X,
   HSCALE=(smin,smax), VSCALE=(xmin,xmax),
   HTITLE='S(m)', VTITLE='X(m)' ;
 DRIFT T=t0 ;
 PLOT SCAT, NONEWPAGE, H=S, V=X,
   HSCALE=(smin,smax), VSCALE=(xmin,xmax),
 DRIFT T=t0+dt ;
 PLOT SCAT, NONEWPAGE, H=S, V=X,
   HSCALE=(smin,smax), VSCALE=(xmin,xmax),
```

with appropriate definitions of t0, smin etc. The DRIFT command transports the beam to the plane t=constant (snap shot). NONEWPAGE operand suppresses page break so that the (s,x) profiles at different times appear on the same page.

η_x, η_y Eta function (m).

η'_x, η'_y Derivative of eta function.

$d\varepsilon/dt$ Coherent energy slope from bunch head to tail (1/m).

ϕ_{xy} Roll angle of the beam in the x-y plane. (radian)

ζ_x,ζ_y,ζ_s Polarization vector. Default=(0,0,0). Note the sign of ζ_s for left-going particles. In the case of photon beams, these are the Stokes parameter (ξ_1, ξ_2, ξ_3). The basis vector of the Stokes parameter is $(\boldsymbol{e}^{(1)}, \boldsymbol{e}^{(2)}, \boldsymbol{e}^{(3)})$

where $\boldsymbol{e}^{(3)}$ is the unit vector along the particle momentum, $\boldsymbol{e}^{(1)}$ is the unit vector along $\boldsymbol{e}_x - \boldsymbol{e}^{(3)}(\boldsymbol{e}^{(3)} \cdot \boldsymbol{e}_x)$, and $\boldsymbol{e}^{(2)} = \boldsymbol{e}^{(3)} \times \boldsymbol{e}^{(1)}$.

See Sec.5.3 for rigorous definitions.

Read particle data from a file

Syntax:

BEAM FILE=f_n|'file_name', [N=N_p,] [NAMELIST,] ;

f_n File reference number.

file_name Existing file name. Must be enclosed by apostrophes. Either full path or relative path. Note that **CAIN** is run in the directory `cain/exec`. The file is opened with the reference number 99 and is closed immediately after reading.

N_p Maximum number of macro-particles to be read in from file. If 0, non-active. Default=0.

NAMELIST FORTRAN NAMELIST format. Othewise the standard format.

Reading file stops when one of the following conditions are satisfied.

- N_p Reached (when $N_p > 0$).
- A file line found whose first three characters are 'END' in the case of the standard format. Or, `END=.TRUE.` found in the case of the NAMELIST format.
- End_of_file detected.

In the case of the standard format, the file is assumed to be created by the following FORTRAN statement.

```
    WRITE(*,'(I2,I6,1X,A4,1P12D20.12)') KIND,GEN,NAME,WGT,
   1      (TXYS(I),I=0,3),(EP(I),I=0,3),(SPIN(I),I=1,3)
```

Here, `NAME` is blanck unless the particle is a test particle or a lost particle or an incoherent-pair particle, `WGT` is the number of real particles expressed by one macro-particle and `GEN` is an integer expressing the generation (1 for the initial particles, 2 for secondaries, etc.). `SPIN` is the polarization vector for electron/positron and the Stokes parameter for photons.

The file can also be MATHEMATICA style (automatically detected). The format string is `('{',I1,',',I5,',',A4,12(',',1PD19.12),'},')`

In the case of the NAMELIST format, the namelist `BEAMIN` must be inserted for each particle.

```
    &BEAMIN
      KIND=2, GEN=1, PNAME='    ', WGT=0.0, TXYS=0.0, 0.0, 0.0, 0.0,
      EP=0.0, 0.0, 0.0, 1.0,  SPIN=0.0, 0.0, 0.0,
      END=.FALSE., SKIP=.FALSE.
    &END
```

Here, the r.h.s. show the number of data, the data type, and the default. The last component of `EP`, i.e., P_s, must not be zero. All the particles must be either right- or left-going. (Actually the particle energy is calculated from $\sqrt{\boldsymbol{p}^2+m^2}$. The input data is not used.)

If the first character of `PNAME` is `T`, the particle is treated as a test particle. (The test particle name must be unique.) `PNAME` should be blanck for normal particles.

If `SKIP=.TRUE.`, the present data is omitted. If `END=.TRUE.`, the present data and all the following data are ignored. Comments in NAMELIST statements follow the local rule on the platform.

To modify the file data (shift of origin, rotation, etc) can be done to some extent by using the command `LORENTZ`.

Test particles

Definition of test particles can also be done by `BEAM` command. One `BEAM` command is needed for each test particle. The number of test particles times the number of `PUSH` time steps must be less than 5000 (parameter `MTSTP` in the file 'cain/source/include/tstpcm.h'). Test particles do not create a field but feel a field. They do not interact with lasers and do not create particles (such as beamstrahlung, incoherent pair, etc). Therefore, 'test photon' does not make sense.

Coordinates and energy-momentum of test particles can be refered to at any time by functions `TestT`, etc. See Sec.2.6.

Syntax:

`BEAM    TESTPARTICLE,   NAME=`n`,|'name',   KIND=`k`,   [TXYS=(`t`,`x`,`y`,`s`),]`
`P=(`p_x`,`p_y`,`p_s`) ;`

n,name — A test particle must have a name, consisting of upto 3 characters. The 'name' (left-adjusted) must be enclosed by a pair of apostrophes. It can also be specified by an integer $-99 \leq n \leq 999$, which is converted to a decimal character string (right-adjusted). Thus, `NAME=1` and `NAME='  1'` is identical. (In the computer, one character '`T`' is added at the top. Thus, `NAME=999` becomes `T999`.)

k — Particle specie.

t,x,y,s — Location of the test particle (m).

p_x, p_y, p_s — 3-momentum (eV/c). p_s must not be zero (i.e., either right-going or left-going).

What is actually defined by the particle variables (t, x, y, s) and (E, p_x, p_y, p_s) is not a particle at a definite space-time coordinate, but rather is a straight trajectory (a world line) which passes the space-time point (t, x, y, s). At the time when the `PUSH` command is executed, they are first pulled to the intercept on $t = t_0$ plane, where t_0 is the starting time of the `PUSH` loop.

When a `BEAM` command is inserted within a `PUSH` loop, the particles are taken to the corresponding time t=`Time`. However, it is safer not to insert `BEAM` command within `PUSH` loop unless you know well what is going on. One exception is the test particles, which in some cases you want to create during a PUSH loop (for example to see the behavior of a low energy particle created during interaction). If you do not want them to be time-shifted in such cases, you can define the TXYS operand as TXYS=(Time,...), where Time is the PUSH running time ('present time').

3.4 LASER

Defines a laser. There can be upto 5 lasers but this can easily be increased (parameter MLSR in source/include/lasrcm.h). One LASER command defines one laser. Note that lasers, if there are more than one, act incoherently. Their interference effects cannot be included in the present version.

The longitudinal (time) pulse shape can be Gaussian or trapezoidal but the transverse is Gaussian only.

Syntax:

LASER [RIGHT|LEFT,] WAVELENGTH=λ_L, POWERDENSITY=P_{peak},
[TXYS=(t,x,y,s),] E3=($e_x^{(3)}$,$e_y^{(3)}$,$e_s^{(3)}$), E1=($e_x^{(1)}$,$e_y^{(1)}$,$e_s^{(1)}$),
RAYLEIGH=(β_1,β_2), [GCUT=n_{cut},] SIGT=σ_τ|TTOT=τ_{tot}, [GCUTT=n_{tcut},]
[TEDGE=τ_{edge},] [STOKES=(ξ_1,ξ_2,ξ_3),] ;

RIGHT|LEFT Right-going or left-going. If RIGHT(LEFT) is specified, the laser acts only onto the left(right)-going particles (to save computing time). If omitted, acts on both.

λ_L Laser wavelength (m).

P_{peak} Peak power density (Watt/m^2).

t,x,y,s Laser focal point and the time when the laser pulse comes there (m).

$(e_x^{(3)}, e_y^{(3)}, e_s^{(3)})$ Unit vector $\boldsymbol{e}^{(3)}$ along the direction of laser propagation.

$(e_x^{(1)}, e_y^{(1)}, e_s^{(1)})$ Unit vector $\boldsymbol{e}^{(1)}$ perpendicular to $\boldsymbol{e}^{(3)}$. $(\boldsymbol{e}^{(1)}, \boldsymbol{e}^{(2)}, \boldsymbol{e}^{(3)})$ with $\boldsymbol{e}^{(2)} = \boldsymbol{e}^{(3)} \times \boldsymbol{e}^{(1)}$ forms a right-handed orthonormal frame. $\boldsymbol{e}^{(3)}$ and $\boldsymbol{e}^{(1)}$ need not be normalized exactly and need not be perpendicular to each other exactly (The component parallel to $\boldsymbol{e}^{(3)}$ is subtracted from $\boldsymbol{e}^{(1)}$ by Schmidt orthogonalization).

β_1,β_2 Rayleigh length in $\boldsymbol{e}^{(1)}$, $\boldsymbol{e}^{(2)}$ direction. (meter)

n_{cut} Cut off of transverse tail in units of sigmas. Default=2.5.

σ_τ R.m.s. pulse length (times velocity of light) in power, not in field amplitude, assuming Gaussian structure. (meter)

n_{tcut} Cut off of longitudinal tail in units of sigmas for Gaussian time structure. Default=2.5.

τ_{tot} Total pulse length for trapezoidal longitudinal structure (meter). Either one of SIGT or TTOT must be specified.

τ_{edge} Longitudinal edge length (meter) for trapezoidal time structure. The flat-top length is then $\tau_{tot} - 2\tau_{edge}$. Default=0 (i.e., rectangular shape).

ξ_1,ξ_2,ξ_3 Stokes parameter defined in the $(\boldsymbol{e}^{(1)}, \boldsymbol{e}^{(2)}, \boldsymbol{e}^{(3)})$ frame. Default=(0,0,0).

See Sec.5.7.1 for more detail.

3.5 LASERQED

Defines the method and parameters for the calculation of the interaction between lasers and particles.

Syntax:

LASERQED COMPTON|BREITWHEELER[,]† NPH=n_{ph}, [NY=n_y,] [NXI=n_ξ,] [NLAMBDA=n_λ,] [NQ=n_q,] XIMAX=ξ_{max}, LAMBDAMAX=λ_{max}, ETAMAX=η_{max}, [PMAX=p_{max},] [ENHANCEFUNCTION=f_{enh},] ;

COMPTON|BREITWHEELER Specifies which parameters to define here.

n_{ph} Maximum number of laser photons to be absorbed in one process.
If < 0, turn off Compton or Breit-Wheeler.
If =0, use linear Compton/Breit-Wheeler formula.
If ≥ 1, use nonlinear formula.
Note that $n_{ph} = 0$ and $n_{ph} = 1$ are different. The former is the limit of $\xi \rightarrow 0$, which contains $n_{ph} = 1$ term only, whereas the latter is a truncation of the exact series with respect to n_{ph}. When $n_{ph} = 0$, none of the variables (n_y, n_ξ, n_λ, n_q, ξ_{max}, λ_{max}, η_{max}) are needed.
When $n_{ph} \geq 1$, only longitudinal polarization is considered and the lasers must be circularly polarized by 100% (i.e., $\xi_1 = \xi_3 = 0$, $\xi_2 = \pm 1$).

n_y Number of abscissa for final energy. Default=20.

n_ξ Number of abscissa for ξ parameter. Default=20.

n_λ Number of abscissa for λ parameter. Applies to Compton case only. Default=20.

n_q Number of abscissa for q parameter. Applies tp Breit-Wheeler case only. Default=50.

ξ_{max} Maximum value of ξ for the table.

λ_{max} Maximum value of λ for the table. Applies to Compton case only.

η_{max} Maximum value of η for the table. Applies to Breit-Wheeler case only.

p_{max} Maximum probability of events per one time step. If the computed probability exceeds p_{max}, **CAIN** of present version stops with a message.

f_{enh} Defines a function in order to enhance a part of spectrum. It is defined as an expression containing Y as the final energy parameter ($0 \leq$Y≤ 1). Its value must be ≥ 1 for $0 \leq$Y≤ 1. Generally speaking, Y close to 1 generates low energy charged particles. For example,

`ENH=1+Step(Y-0.8)*(Y-0.8)*10`
will enhance the events with `Y`$>$ 0.8 by a factor upto 3 (at `Y`=1).
In the program, the real spectrum function is multiplied by f_{enh} and, when an event is generated, the created particles are asigned a weight $1/f_{enh}$.
Note that f_{enh} slightly larger than 1 is useless (even harmful) because a small fraction $1-1/f_{enh}$ of the parent particle will remain as a macro-particle, causing a waste of computing time. In the example above, $f_{enh}=1$ exactly for `Y`<0.8.
This function is used only during the initialization by `LASERQED` command. Therefore, if the expression contains user-defined parameters, their values at the time of `LASERQED` command are used. Changing them afterwards will not affect the computation.

See Sec.5.7.3 and Appendix.A for more detail

3.6 CFQED

Constant-Field QED, i.e., the beamstrahlung and coherent pair creation. Both the effects of the beam field and the external field are included. The angular distribution of the final particles is not included.

When the polarization flag (see below) is on, all the polarization effects (longitudinal and transverse spin of electron/positron and linear and circular polarization of photon) are included.

Syntax:

`CFQED` `BEAMSTRAHLUNG`|`PAIRCREATION`[,]† [`POLARIZATION`,]
[`PMAX=`p_{max},] [`WENHANCE=`w_{enh},] ;

`BEAMSTRAHLUNG`|`PAIRCREATION` Specifies which parameters to define here. Only one of these may be specified by one `CFQED` command.

`POLARIZATION` Flag to take into account all the polarization effect. (default=No). Note that the flag `SPIN` (`FLAG` command) must also be on for polarization calculation.

p_{max} Maximum probability of events per one time step. (Default=0.1). When the probability exceeds p_{max}, **CAIN** stops with a message.

w_{enh} Enhancement factor of radiation rate. $0 \le w_{enh}$. When $w_{enh}=1$ (default), macro-photons are created such that $n_{\mathrm{macro}\gamma}/n_{\mathrm{macroe}} = n_{\mathrm{real}\gamma}/n_{\mathrm{reale}}$
When $w_{enh}>1$ (<1), macro-photons are created more (less) by the factor w_{enh}, each having less (more) weight. When $w_{enh}=0$, no photon is created (but the recoil of electron is taken into account.) This operand is introduced in order to avoid poor statistics due to too less macro-photons or memory overflow due to too many macro-photons.

See Sec.5.8 and Sec.5.9 for the formulas and algorithm and for more detail on the enhancement factor.

3.7 BBFIELD

Define the parameters for the calculation of beam-beam field.
Syntax:

BBFIELD WX=w_{x1}|WX=(w_{x1}[,w_{x2}]), [WXMAX=w_{xm1}|WXMAX=(w_{xm1},w_{xm2}),]
R=r, [NX=n_x,] [NY=n_y,] [PSIZE=Δ,] [NMOM=n_{mom},] ;

w_{x1},w_{x2} Horizontal width of the mesh in meters for right and left-going beams. If w_{x2} is not specified, $w_{x2} = w_{x1}$ is adopted. No default for w_{x1}.

w_{xm1},w_{xm2} If WXMAX is given, the with of the mesh region can vary in the range (w_x, w_{xm}) when the beam fraction outside the range defined by WX and R is significant. Note $w_{xm} \geq w_x$.

r Aspect ratio $(w_x/n_x)/(w_y/n_y)$ of the horizontal to vertical mesh size. This is common to right and left-going beams. No default.

n_x,n_y Number of horizontal and vertical bins. Present version uses Fast Fourier Transformation so that a power of 2 is the best choice. Other numbers are also allowed but those of the form 2^n or 3×2^n or 5×2^n are recommended. Default=32.

Δ Macro-particle size in units of the bin size. Macro-particles are treated as a rectangular of uniform distribution. Must be $0 \leq \Delta \leq 1$. Default=1.

n_{mom} For (x, y) points outside the mesh region, a harmonic expansion using the elliptic coordinate is used. The parameter n_{mom} specifies the truncation of harmonics. $n_{mom} = 0$ takes only the total charge term and $n_{mom} < 0$ ignores the field outside. Default=10.
Note that the particles outside mesh region receive the beam-beam kick unless $n_{mom} < 0$, but the field created by them is not taken into account. See Sec.5.6 for more detail.

Note that the longitudinal mesh size, which is common to beam-beam field and luminosity calculations, has to be defined by the parameter Smesh by the SET command. Its value at the time when PUSH started is used thoughout the PUSH loop.

3.8 EXTERNALFIELD

Define external field. The present version allows only a constant field over an interval bordered by two parallel planes.
Syntax:

EXTERNALFIELD [S=(s_1,s_2),] [V=(c_x,c_y,c_s),] [E=(E_x,E_y,E_s),]
[B=(B_x,B_y,B_s),] ;

s_j, c_j Define the range of the field as $s_1 \leq c_x x + c_y y + c_s s \leq s_2$.
Must be $s_1 < s_2$. Default $s_1 = -\infty$, $s_2 = +\infty$ and (c_x,c_y,c_s)=(0,0,1).

E_j Electric field components in units of V/m. Default=(0,0,0).

B_j Magnetic field components in units of Tesla. Default=(0,0,0).

3.9 LUMINOSITY

Define the transverse mesh size, number of bins, etc, for luminosity calculation. One luminosity command is needed for each combination of particles γ, e^-, e^+, right-going and left-going. Thus, there can be at most 9 `LUMINOSITY` commands.

Syntax:

```
LUMINOSITY    KIND=(k_r,k_l),    [FREP=f_rep,]
  [W=(W_min,W_max,n_bin),|W=(W_0,W_1,...,W_n_bin),]
  [E1=(E_1min,E_1max[,n_1bin]),|E1=(E_1,0,E_1,1,...,E_1,n_1bin),]
[E2=(E_2min,E_2max[,n_2bin]),|E2=(E_2,0,E_2,1,...,E_2,n_2bin),]   WX=(w_x[,w_xm]),
  WY=(w_y[,w_ym]),   [HELICITY,]   [ALLPOL,] ;
```

k_r,k_l Particle species of right and left-going beams.

f_{rep} Repetition frequency (Hz). Used for the luminosity scale only. Default=1Hz.

W_{min},W_{max},n_{bin} Parameters for differential luminosity with respect to the center-of-mass energy W. (W_{min},W_{max}) is the range in eV and n_{bin} is the number of bins. If (W_{min},W_{max}) is not given, the center-of-mass spectrum is not calculated. Default for n_{bin} is 50.

W_j,n_{bin} Define the center-of-mass energy bins in the case of non-equal spaced bins. n_{bin} is the number of bins, W_0 is the lower edge of the first bin and $W_{n_{bin}}$ is the upper edge of the last bin. n_{bin} must be ≥ 3 in order to distinguish from the equal-space case.

E_{1min},E_{1max},n_{1bin},E_{2min},E_{2max},n_{2bin} 2-D differential luminosity $d\mathcal{L}/dE_1 dE_2$. (E_{jmin}, E_{jmax}) is the range in eV and n_{jbin} is the number of bins. (j=1 for right-going beam and j=2 for left-going beam.) Both or none of `E1` and `E2` have to be specified. If none is specified, 2-D luminosity is not calculated. Default for ni, bin is 50.

$E_{i,j}$,$n_{i,bin}$ $(i = 1, 2)$ Define non-equal spaced bins. Similar to the case of the center-of-mass energy.

w_x,w_y Full horizontal/vertical width of the mesh region (m). The origin is adjusted automatically from time to time.

w_{xm},w_{ym} Maximum width of the mesh region (m). If not given, $w_x(w_y)$ is used throughout. If given, an increased size upto $w_{xm}(w_{ym})$ is used when a significant particle fraction gets out of the mesh region defined by (w_x, w_y). The number of mesh points is determined automatically.

`HELICITY` Calculate luminosity for every combination of helicity, (++), (−+), (+−), (−−).

`ALLPOL` Calculate luminosity for all possible 16 combinations of the spins. (see Sec.5.5.2 for detail.)

All the LUMINOSITY commands must have the same value of w_x,w_y,w_{xm},w_{ym}, and f_{rep}. (Specify them at the first LUMINOSITY command.)

Note that the longitudinal mesh size, which is common to beam-beam field and luminosity calculations, has to be defined by the parameter Smesh by the SET command.

The luminosity is actually computed by the PUSH-ENDPUSH loop. The calculated luminosity can be referred to by the following functions. (If during the loop, the accumulated luminosity upto that moment is returned.)

Lum(k_r,k_l) Luminosity of KIND=(k_r,k_l) in units of cm^{-2}sec^{-1}.

LumH(k_r,k_l,h) Helicity luminosity: helicity combination $(++)$ $(h=1)$, $(-+)$ $(h=2)$, $(+-)$ $(h=3)$, $(--)$ $(h=4)$. $h=0$ will give the total luminosity Lum(k_r,k_l). (cm^{-2}sec^{-1})

LumP(k_r,k_l,s_1,s_2) Polarization luminosity. ($0 \le s_1 \le 3$, $0 \le s_2 \le 3$) See Sec.5.5.2 for definition. (cm^{-2}sec^{-1})

LumW(k_r,k_l,n) Differential luminosity in the n-th bin. (cm^{-2}sec^{-1}/bin)

LumWbin(k_r,k_l,n) Bin center (eV) of the n-th bin. If $n=0$, the number of bins is returned. (Error if $n<0$ or n is larger than the number of bins.)

LumWbinEdge(k_r,k_l,n) Bin edge (eV) of the n-th bin. ($0 \le n \le$ number of bins. $n=0$ is the lower edge of the first bin and n =number of bis is the upper edge of the highest bin. (Error if $n<0$ or n is larger than the number of bins.)

LumWH(k_r,k_l,n,h) Differential helicity luminosity. (cm^{-2}sec^{-1}/bin)

LumWP(k_r,k_l,n,s_1,s_2) Differential polarization luminosity. ($0 \le s_1 \le 3$, $0 \le s_2 \le 3$) See Sec.5.5.2 for definition. (cm^{-2}sec^{-1}/bin)

LumEE(k_r,k_l,n_1,n_2) 2-D differential luminosity $d\mathcal{L}/dE_1dE_2$ for the bin (n_1,n_2). (cm^{-2}sec^{-1}/bin)

LumEEbin(k_r,k_l,l,n) Bin center (eV) of the n-th bin of E_1 (l=1) or E_2 (l=2). If $n=0$, the number of bins is returned. (Error if $n<0$ or n is larger than the number of bins.)

LumEEbinEdge(k_r,k_l,l,n) Bin edge of the n-th bin of E_1 (l=1) or E_2 (l=2). See LumWbinEdge for the definition of n.

LumEEH(k_r,k_l,n_1,n_2,h) 2-D differential helicity luminosity

LumEEP(k_r,k_l,n_1,n_2,s_1,s_2) 2-D differential polarization luminosity.

These functions can be included in expressions. Thus, you can write the computed luminosity on a file. In particular, the only way to retrieve the 2-D luminosity $d\mathcal{L}/dE_1dE_2$ is to use the above functions because PLOT LUMINOSITY command cannot plot it (KEK TopDrawer cannot draw 3-D plot). So, for example, to write e$^+$e$^-$ luminosity, say

```
SET m1=LumEEbin(2,3,1,0), m2=LumEEbin(2,3,2,0);
WRITE ((LumEE(2,3,n1,n2),n1=1,m1),n2=1,m2),
  FORMAT=(...);
```

If you are satisfied with a pre-defined format, you can use PRINT/WRITE LUMINOSITY command.

3.10 PPINT

Incoherent particle-particle interaction such as incoherent pair creation and bremsstrahlung. The following processes are included:

Breit-Wheeler	$\gamma + \gamma \to \mathrm{e}^- + \mathrm{e}^+$
Bethe-Heitler	$\gamma + \mathrm{e}^\pm \to \mathrm{e}^\pm + \mathrm{e}^- + \mathrm{e}^+$
Landau-Lifshitz	$\mathrm{e} + \mathrm{e} \to \mathrm{e} + \mathrm{e} + \mathrm{e}^- + \mathrm{e}^+$
Bremsstrahlung	$e + e \to e + e + \gamma$

All the processes except for Breit-Wheeler are cvalculated using the virtual photon approximation.

The circular polarization effect of the initial photons is included in the Breit-Wheeler process but all other polarization effects are ignored.

Particles created by incoherent processes do not contribute in creating the beam field. Also note that the parent macro-particles do not change by particle-particle interaction. All these come from the actual situation in linear colliders where the incoherent particles are much less in number compared with the initial particles.

Syntax: Specify virtual photon options

```
PPINT    VIRTUALPHOTON[,]†   [LOCAL,] [FIELDSUPPRESSION,] [EMIN=Emin,] ;
```

LOCAL Flag to adopt local virtual photon, i.e., to ignore the effects due to the finite transverse extent of virtual photons. Default is non-local.

FIELDSUPPRESSION Flag to include the virtual-photon suppression effect due to strong external fields (normally, the beam-field by the on-coming beam). This can be effective when LOCAL is not specified. See section 3.4 of [6]. Default: does not include this effect.

E_{min} Minimum energy of final electron/positron energies in eV. Default is twice the rest mass of electron. = 1.022...E6. This parameter is not directly related to virtual photons but included here because it is common to all the processes. The purpose of this parameter is to save computing time. The creation of pairs does not take too much computing time but to track extremely low-energy pairs in a strong beam field is very expensive. The worst ones are the pair particles having the sign of charge opposite to that of the on-coming beam because they are trapped in the strong field region. If you are not interested in them, you can eliminate them during the PUSH loop as

```
CLEAR BEAM, INCP, RIGHT, KIND=2;
CLEAR BEAM, INCP, LEFT, KIND=3;
```

if the right(left)-going beam is electron(positron).

Syntax: Specify individual processes

```
PPINT    BW|BH|LL|BREMSSTRAHLUNG[,]†   [RIGHT,]   [LEFT,]
  [ENHANCE=fenh,] ;
```

BW,BH,LL,BREMSSTRAHLUNG Specify one of Breit-Wheeer, Bethe-Heitler, Landau-Lifshitz, and Bremsstrahlung interactions. If more than one of these are needed, apply PPINT command repeatedly. No default.

RIGHT,LEFT Applies to Bethe-Heitler and Bremsstrahlung. The Bethe-Heitler process has two possible combinations, namely, (γ,e$^\pm$) and (e$^\pm$,γ). RIGHT/LEFT option specifies the photon is right-going or left-going or both. Default is both.
The Bremsstrahlung is treated as the interaction between real e$^\pm$ and a virtual photon. Therefore, it also has two possible combinations. This operand specifies which beam is the real particle.

f_{enh} Event rate enhancement factor. It is unity when the number of created macro-pairs is the same as the expected number of real pairs, *i.e.*, the weight of the pair particle is $1/f_{enh}$. Default f_{enh}=0.1.

In using **ABEL** one had to define the minimum scattering angle and minimum transverse momentum. This was due to the ultra-relativistic approximation employed there. **CAIN** does not need these parameters.

3.11 PUSH, ENDPUSH

Define the time step loop of tracking. Tracking is done by a pair of commands instead of one single command in order to allow users to take action such as print, plot, insert test particles, etc, at arbitrary time steps.
Syntax:

PUSH Time=(t_0,t_f,n_t) ;

... any commands ...

ENDPUSH ;

t_0,t_f Start and end time (multiplied by velocity of light) of tracking (meter). Note the spelling of Time which contains lower case alphabet in contrast to other operand keywords consisting of upper case letters only. In fact, Time is a pre-defined variable name. Therefore, you can, for example, print its current value during PUSH loop by PRINT Time, FORMAT=....

n_t Number of time steps. (≥ 0)

Actual control of the loop is done in the following way.

- Before the first time step, all the particles are made to drift to $t = t_0$ (by straight lines).
- At the PUSH command of j-th loop ($j = 0, 1, \ldots, n_t$), the time variable Time is set to $t_j = t_0 + j\Delta t$ where $\Delta t = (t_f - t_0)/n_t$.
- Execute commands between PUSH and ENDPUSH.
- Control comes to ENDPUSH. If $j < n_t$, make tracking (beam-beam, beamstrahlung, laser interaction etc) for the time step $t_j \leq$ Time $\leq t_{j+1}$.

- If $j < n_t$, returns to PUSH.

Note that the commands between PUSH and ENDPUSH are executed $n_t + 1$ times. If $n_t = 0$, the actions taken are to drift all particles to t_0 and to do commands between PUSH and ENDPUSH once. If $n_t < 0$, **CAIN** stops at PUSH with an error message rather than at ENDPUSH.

3.12 DRIFT

Drift the particles to a certain time or to a certain s coordinate.
Syntax:

DRIFT T=t_1|DT=Δt|S=s_1, [RIGHT,] [LEFT,] [KIND=k|(k_1,k_2),]
[EXTERNALFIELD,] ;

t_1 Drift until Time=t_1 (meter).

Δt Drift over time interval Δt (meter).

s_1 Drift to s coordinate $= s_1$ (meter).
In any of the three cases T, DT, and S, the particles may go backwards in time depending on the parameters.

RIGHT,LEFT Drift right- or left-going particles only.

k Drift only particles of kind k.

EXTERNALFILED Take into account the external field.

When there is only external field without beam interaction, DRIFT EXTERNAL is much better (more accurate and faster) than the PUSH command. The difference is that DRIFT EXTERNAL uses an exact solution in a constant field whereas PUSH carries out step-by-step integration, and that PUSH accepts only t as the independent variable while DRIFT also allows s (as in most accelerator program codes).

How to use DRIFT EXTERNAL may be understood by the following example. Suppose that the region $s_1 < s < s_2$ is shined by a laser. An electron beam comes from the left and goes through the laser region to created back-scattered photons, and subsequently goes through a magnetic field region $s_3 < s < s_4$. If the interval (s_2, s_3) is shorter than the bunch length, the bunch head is already in the field region when the tail gets out of the laser region. If you use PUSH command, you have to track the beam till the end of the magnetic field region. Instead, you can do more elegantly,

BEAM	Define electron beam
LASER	Define laser
LASERQED	Define laser QED parameters
PUSH	Start push without magnetic field
ENDPUSH;	End push
EXTERNALFIELD	Define external field
DRIFT S=s_3;	Pull back the beam to the plane s_3
DRIFT S=s_4, EXTERNALFIELD;	Calculate the field effects

3.13 LORENTZ

Coordinate transformation (shift of origin, rotation, Lorentz transformation) of particle coordinate, energy-momentum, polarization, etc. Using this command, you can transform a collision at an angle into a head-on collision.

Syntax:

LORENTZ [TXYS=(Δt,Δx,Δy,Δs),] [ANGLE=ϕ,] [BETAGAMMA=β_γ,]
[AXIS=(a_x,a_y,a_s),] [EV=(e_{vx},e_{vy},e_{vs}),] [NOBEAM,] [RIGHT,] [LEFT,]
[KIND=k|(k_1,k_2),] [EXTERNALFIELD,] [LASER,] ;

(Δt, Δx, Δy, Δs) Shift of origin. (m)

ϕ Spacial rotation angle (radian). (rotation of the coordinate axis.)

β_γ Lorentz boost parameter $\beta \times \gamma$. (Boost of the coordinate axis).

(a_x, a_y, a_s) Unit vector along the rotation axis. Need not be normalized exactly.

(e_{vx}, e_{vy}, e_{vs}) Unit vector along the boost direction. Need not be normalized exactly.

NOBEAM No transformation of particles. If specified, RIGHT, LEFT, KIND operands are ignored.

RIGHT,LEFT Select right- or left-going particles only. If omitted, both are transformed.

k Select only particles of kind k. If omitted, all species are transformed.

EXTERNALFIELD Lorentz transformation of external field (transformation of the field strength and the boundary).

LASER Lorentz transformation of lasers.

The three types of transformations are carried out in the order of the input keywords TXYS, ANGLE, BETAGAMMA. With one LORENTZ command, each transformation can be specified at most once.

Note that, for any type, the transformation is that of the coordinate axis rather than the particles themselves. Thus, for example, if you say TXYS=(0,0,0,1), then the s-coordinate of the particles decreases by 1 meter.

3.14 DO, ENDDO

Do loop. Can be nested. Two forms are possible.

Syntax: form-1

DO REPEAT[,]† n ;

n Number of repetition. Can be an expression (evaluated when entering the loop). $n > 0$. ($n = 0$ causes a jump to ENDDO. $n < 0$ causes an abnormal term.)

Syntax: form-2

```
DO    WHILE[,]†   a   rel   b ;
```

a, *b*   Expressions.

rel   A relational operator. One of =, <, >, <=, >=, =<, =>, <>, ><.

The loop is repeated so long as the condition is satified. The check is made at the time of `DO` command. The values of expressions are REAL*8. If you want integers for definiteness, use `Nint( )` or `Int( )`. End of do loop is

Syntax:

```
ENDDO ;
```

Do not forget ";".

3.15 IF,ELSE, ENDIF

Define if block. Can be nested. Note that 'THEN' is not needed. The `ELSE` clause may be absent. More complicated forms of logical expressions including 'and'/'or' are not ready yet.

Syntax:

```
IF    a   rel   b ;
............ ;
ELSE ;
............ ;
ENDIF ;
```

a, *b*   Expressions.

rel   A relational operator. One of =, <, >, <=, >=, =<, =>, <>, ><.

Do not forget ";".

3.16 WRITE, PRINT

Write some data. The only difference between `WRITE` and `PRINT` is the default destination which is `OutFile` for `WRITE` and `MsgFile` for `PRINT`. Therefore, they are identical if `FILE` operand is specified. Another difference is that errors in reading the command cause abnormal termination for `WRITE` whereas the command is ignored for `PRINT`.

Write the macro-particle data

Syntax:

```
WRITE    BEAM,   [FILE=fn|'file_name',]   [APPEND,]   [RIGHT|LEFT,]
  [KIND=k,]   [INCP,]   [SHORT|MATHEMATICA] ;
```

BEAM Write beam data.

f_n File reference number. See above for default.

file_name New or old file name. Must be enclosed by apostrophes. Either full path or relative path. Note that **CAIN** is run in the directory `cain/exec`. When file_name is specified, the input file reference number is ignored and the file is opened with the reference number 98.

APPEND Append in an existing file. (Ignored if does not exist.)

SHORT Short format which fits to a wide screen.

METHEMATICA MATHEMATICA style format. Use standard format (see Sec.3.3) if none of the avobe two are specified.

RIGHT|LEFT Write only either right-going or left-going particles. Default=both.

INCP Write particles created by incoherent processes (defined by `PPINT` command). Otherwise, normal particles only. If you want both, execute the command twice.

k Write only photon ($k = 1$) or electron ($k = 2$) or positron ($k = 3$) selectively. Default=all.

Write the beam statistics data

Syntax:

```
PRINT    STATISTICS,   [INCP,] [SHORT|LONG,]   [FILE=fn,]   [APPEND,] ;
```

STATISTICS Write beam global data such as number of particles, r.m.s. size, etc.

SHORT Print only the number of macro- and real particles. If none of `SHORT` and `LONG` is specified, print average and r. m. s. of (t, x, y, s) and (E, p_x, p_y, p_s) as well as the average spin components.

LONG Print max. and min. in addition to the standard items.

INCP Include incoherent particles only. Otherwise, normal particles only. If you want both, execute the command twice.

f_n File reference number. See above for default.

Write the calculated luminosity

Syntax:

```
PRINT    LUMINOSITY,   KIND=(k1,k2)   [FILE=fn,]   [APPEND,] ;
```

LUMINOSITY Write calculated luminosity specified by (k_1,k_2).

k_1,k_2 Define right and left-going beams. All the luminosities (differential and polarization) defined by the `LUMINOSITY` command will be printed. The print format is complicated. Just try.

f_n File reference number. See above for default.

Write the values of parameters and expressions

Syntax:

`PRINT` [`PARAMETER,`] [`FILE=`f_n`,`] x_1[, x_2[, $x_3 \ldots$]], [`FORMAT=(`fmt`),`] ;

`PARAMETER` Write values of (predefined or user defined) parameters or expressions. Can be omitted.

f_n File reference number. See above for default.

x_j Expressions. It is safer to enclose each expression by `( )` or `[ ]` or `{ }`.[3] It is also possible to write a do-type sequence of the form (almost like FORTRAN)

(y_1,...,y_n,i=i_1,i_2,i_3)

where y_j's are expressions, i is a user-parameter name (need not be defined by `SET` command), i_1, i_2, and i_3 are expressions for initial, final, and increment values of i. If i_3 is omitted, $i_3 = 1$ is adopted. Note that i_1, i_2, and i_3 are considered to be integers. (`Nint` is applied.)

Do-type sequence may be nested as in FORTRAN. The do-control variable must not duplicate, of course. (Duplication within the sequence is checked but possible interference with variables outside `PRINT` or `WRITE` is not checked.)

(fmt) Fortran format. Must be enclosed by `( )`. **CAIN** does not check the grammer so that a wrong format will cause an abnormal term by your computer system. Note that all the expressions are REAL*8. You cannot use I-format. If format is not specified, printed as 'expression=value' by 1PD15.8 (one line for each).

If format is given but there is no expression to be printed, the format is executed as in FORTRAN. For example,

```
WRITE FORMAT=('nothing');
```

will cause '`nothing`' be printed.[4]

Unfortunately, the grammer of **CAIN** does not allow an un-paired apostrophe so that, for example, `1H'` will cause an error.

[3] There is no such a rule that a user parameter name must not be identical to some keyword. Here, however, there is an inconsistency of grammer. If you define a parameter with the name `ST`, for example, `PRINT ST` may be understood as printing the parameter or printing the statistics, unless the keyword `PARAMETER` is explicitly written. This can be avoided by writing `PRINT (ST)` because `(ST)` is not a keyword but is an expression actually identical to `ST`.

[4] A known bug. `WRITE (i=1,2), FORMAT=('nothing');` will not work as you expect in FORTRAN.

3.17 PLOT

Plot using TopDrawer.

Histogram of particle data

Syntax:

> PLOT HIST, [NONEWPAGE,] [RIGHT|LEFT,] [KIND=k|(k_1,k_2),]
> [GENERATION rel n,] [INCP,] H=f_x, [HSCALE=(x_{min},x_{max},n_{bin}),]
> [HLOG,] [VLOG,] [TITLE='head_title',] [HTITLE='bottom_title',]
> [VTITLE='left_title',] [FILE=f_n,] ;

NONEWPAGE Do not insert 'NEWFRAME' of TopDrawer so that the figure is written on the previous plot on the same file. This makes sense when the new plot has the same scale as the previous plot.

RIGHT|LEFT Select right(left)-going particles only.

k,k_1,k_2 Select photons ($k = 1$), electrons ($k = 2$), positrons ($k = 3$) only.

rel Relational operator. One of =, <, >, <=, >=, =<, =>, <>, ><.

n Generation. Select particles whose generation satisfies the relation "n_{gen} rel n".

INCP Include particles created by incoherent incoherent processes only. Otherwise normal particles only.

f_x An expression defining the horizontal variable. Following running variables can be used. (See Sec.2.5)
T, X, Y, S, En, Px, Py, Ps, Sx, Sy, Ss, Xi1, Xi2, Xi3, Kind, Gen
For example,
`H=Sqrt[(Px^2+Py^2)/Ps^2]*1E6`
defines the orbit angle in micro-radians.

x_{min},x_{max},n_{bin} Minimum and maximum of the horizontal scale and the number of bins. If omitted, the minimum and maximum in the particle data are used for x_{min},x_{max} and $n_{bin} = 50$.

HLOG,VLOG Log scale of horizontal and vertical axes. When HLOG is specified, x_{min} and x_{max} must be specified explicitly. The binning interval will be equal in log-scale.

top_title, etc Title string. Must be enclosed by a pair of apostrophes. Topdrawer case string can be specified by using ";" as the delimitor like
TITLE='EOG1; XGX;', for writing E_γ. It is recommended to put ";" also at the end, as in this example, to avoid writing unnecessary blanck characters.

f_n Output file reference number. Default=TDFile.

Scatter plot of particle data

Syntax:

> PLOT SCATTER, [NONEWPAGE,] [RIGHT|LEFT,] [KIND=k|(k_1,k_2),]
> [GENERATION rel n,] [INCP,] H=f_x, V=f_y, [HSCALE=(x_{min},x_{max}),]
> [VSCALE=(y_{min},y_{max}),] [HLOG,] [VLOG,] [MAXNP=n_{max},]
> [TITLE='top_title',] [HTITLE='bottom_title',] [VTITLE='left_title',]
> [FILE=f_n,] ;

f_y An expression defining the vertical variable.

y_{min},y_{max} Minimum and maximum of the vertical scale.

n_{max} Maximum number of points to be plotted (in order to save the plotting time). Randomly selected. Default: plot all points.

Other operands are the same as for the histogram.

Plot the test particle data

Syntax:

> PLOT TESTPARTICLE, [RIGHT|LEFT,] [KIND=k|(k_1,k_2),] H=f_x,
> V=f_y, [HSCALE=(x_{min},x_{max}),] [VSCALE=(y_{min},y_{max}),]
> [TITLE='top_title',] [HTITLE='bottom_title',] [VTITLE='left_title',]
> [FILE=f_n,] ;

Other operands are the same as for the scatter plot. Note that the information of the test particle history is stored (in contrast to normal particles). Thus, you can say, for example, `H=T` to see the trajectory as a function of time.

The plot may show apparently unphysical features when you apply `DRIFT` command. `DRIFT` command may be used to pull particles to a certain position or time. This does not corresponds to a physical motion. Even in such cases, test particle coordinates are stored at the end of `DRIFT` command. Moreover, in contrast to the `PUSH` command, step-by-step information of test particles during `DRIFT` command is not stored because `DRIFT` command calculates particle trajectories by a single step using exact analytic formulas.

Plot the differential luminosity

The differential luminosity w.r.t. the center-of-mass energy can be plotted if defined by `LUMINOSITY` command and calculated by `PUSH` command. Only the 1-D differential luminosity $d\mathcal{L}/dW$ is plotted. 2-D differential luminosity $d\mathcal{L}/dE_1dE_2$ is not plotted because the TopDrawer available at KEK HP station is not capable of 3-D plot.

Syntax:

> PLOT LUMINOSITY, KIND=(k_1,k_2), [FILE=f_n,] [VLOG,]
> [PERBIN|PERHVAR,] ;

k_1,k_2 Define right and left-going beams. When `HELICITY` operand has been specified in the `LUMINOSITY` command, all the 5 spectrums (unpolarized and 4 combinations of helicities) come out in 5 separate plots.

VLOG Log scale of vertical axis. The horizontal axis cannot be log-scale.

PERBIN Luminosity per bin (1/cm^2/sec/bin) is plotted.

PERHVAR Luminosity per unit increment of horizontal axis (energy) is plotted. (1/cm^2/sec/eV). Default is PERBIN.

More flexible plot is possible with the complicated syntax

Syntax:

PLOT LUMINOSITY, KIND=(k_1,k_2), V=f, [NONEWPAGE,]
[VSCALE=(y_{min},y_{max}),] [VLOG,] [PERBIN|PERHVAR,]
[TITLE='top_title',] [HTITLE='bottom_title',] [VTITLE='left_title',]
[FILE=f_n,] ;

f Defines what is plotted. You can use the following variables.
L0: unpolarized luminosity.
Ln: n=1,2,3,4. helicity luminosity.
Lij: i,j=0,1,2,3. general polarization luminosity.
These are in units of 1/cm^2/s/bin. (Or 1/cm^2/s/eV if PERHVAR is specified.) Ln (Lij) is allowed when HELICITY (ALLPOL) has been specified in LUMINOSITY command.

The operands KIND, PERBIN, PERHVAR are the same as in the first syntax. The rest is the same as in PLOT SCATTER except for V=f. The titles are automatically created in the first syntax but not in the second.

Plot charge distribution and beam-beam field

The charge distribution and the beam field data for beam-beam interaction are computed at each time step for each longitudinal slice but they are not kept in the memory. They can be plotted only at the time moment and for the slice which is being proccessed. Thus, this command is to be inserted during PUSH loop. The slice is specified by the S operand.

Syntax:

PLOT BBFIELD, S=s_1|S=(s_1,s_2,...), [FILE=f_n,] ;

s_j Define the s-coordinate. Plot for the slice which contains one of s_j's. Upto 5 s_j's can be specified.

Plot a function

Syntax:

PLOT FUNCTION, [NONEWPAGE,] H=f_x, V=f_y, PARAMETER=name,
RANGE=(x_1,x_2[,n]), [XLOG,] [HSCALE=(x_{min},x_{max}),]
[VSCALE=(y_{min},y_{max}),] [HLOG,] [VLOG,] [LINEMODE=(l_1,l_2,,,...),]
[TITLE='top_title',] [HTITLE='bottom_title',] [VTITLE='left_title',]
[FILE=f_n,] ;

f_x,f_y Define the function to be plotted in the parameterized form. They should normally contain the variable defined by the `PARAMETER` command.

name Name of the parameter to vary. Must satisfy the constraints as a user-defined variable and must not be the pre-defined names.
If you want to plot the sine function, for example, you would say

```
PLOT FUNCTION, PARAMETER=x, RANGE=(0,2*Pi),
H=x, V=Sin(x), HSCALE=(0,2*PI), VSCALE=(-1,1) ;
```

x_1,x_2 Define the range of the parameter. ($x_1 \neq x_2$)

n Number of points (-1) in the range (x_1, x_2). (Default $n = 100$.)

`XLOG` Divide the range uniformly in log scale. Otherwise linear.

l_1,l_2,... Define the line mode, meaning a line segment of length l_1 (in units of inches) followed by a space of length l_2, followed by a line l_3, etc. The whole pattern is repeated. For example, `LINEMODE = (0.1,0.1)` will cause a dashed line. If `LINEMODE` is not defined or only l_1 is specified, a solid line is plotted.

Other operands are the same as for the histogram. You can plot many functions in a frame by using `NONEWPAGE` option.

3.18 CLEAR

Clear/disable the beam, laser, etc. The first operand is a positional keyword.

Clear particles

Syntax:

```
CLEAR    BEAM[,]†  [TESTPARTICLE,]  [INCP,]  [RIGHT|LEFT,]
  [KIND=k|(k1,k2),] ;
```

`TESTPARTICLE` Clear test particles.

`INCP` Clear particles created by incoherent processes (defined by `PPINT`). If none of `TESTPARTICLE` and `INCP` is specified, normal particles are eliminated. Therefore, if you want to eliminate all, you need `CLEAR` command twice:

```
CLEAR BEAM, TESTPARTICLE, INCP;
CLEAR BEAM;
```

`RIGHT`,`LEFT` Clear right or left-going particles only. (Default=both).

k,k_1,k_2 Clear photon ($k = 1$) or electron (2) or positron (3) only.

Turn off lasers

Syntax:

```
CLEAR    LASER[,]† ;
```

Turn off LASERQED

Syntax:

```
CLEAR    LASERQED[,]†   [COMPTON,]   [BREITWHEELER,] ;
```

Clear parameters for the laser QED. If either one of COMPTON or BREITWHEELER is specified, the other one is not turned off.

Clear luminosity

Syntax:

```
CLEAR    LUMINOSITY[,]† ;
```

Clear luminosity integrals as well as the definitions of luminosities. Note that the contents of luminosity integrals are cleared whenever the PUSH command starts. Thus, if you do another PUSH without CLEAR LUMINOSITY, the luminosity command will be still active and the integration starts from scratch..

Turn off beam-beam field

Syntax:

```
CLEAR    BBFIELD[,]† ;
```

Turn off the external field

Syntax:

```
CLEAR    EXTERNALFIELD[,]† ;
```

Turn off CFQED

Syntax:

```
CLEAR    CFQED[,]†   [BEAMSTRAHLUNG,]   [COHERENTPAIR,] ;
```

If none of BEAMSTRAHLUNG and COHERENTPAIR is specified, both is turned off.

Turn off PPINT

Syntax:

```
CLEAR    PPINT[,]† ;
```

Turn of particle-particle interaction defined by PPINT command. Note that this does not mean to eliminate particles already created.

3.19 FILE

Open, close, rewind a file. The first operand is positional.
Syntax:

FILE OPEN|CLOSE|REWIND|ENDFILE[,]† UNIT=n_f, [STATUS='status',]
[NAME='fname',] ;

n_f Logical file number. No default.

status For OPEN, one of NEW, OLD, SCRATCH, UNKNOWN. Default=UNKNOWN. For CLOSE, one of KEEP, DELETE. Default=KEEP. Not used for REWIND and ENDFILE. Need not be enclosed by apostrophes.

fname File name. Used for OPEN only. Need not be enclosed by apostrophes if the name does not include parenthesis, comma, and semicolon.

3.20 HEADER

Define the header for TopDrawer plots. Effective until next HEADER command appears.
Syntax:

HEADER 'header_string' ;

header_string Character string. Upto 120 characters. Strings delimited by commas like 'string$_1$','string$_2$',... are concatenated. The 'case' string for TopDrawer can be defined by using semicolon ";" as delimiter, like

```
 HEADER 'JLC E0CM1=500GeV;      X   X         ' ;
```

If header_string is not written, the header is cleared.

3.21 STORE and RESTORE

Store/restore the current variables or the luminosity data in/from a file.
Syntax:

STORE [LUMINOSITY,] [FILE='fname',] ;

Syntax:

RESTORE [LUMINOSITY,] [FILE='fname',] ;

LUMINOSITY Store/restore luminosity data. If not specified, store/restore variables.

fname File name. Need not be enclosed by apostrophes if the name does not include parenthesis, comma, and semicolon.
The file is opened with the unit number 98 and is closed after reading/writing. If the file name is omitted, the standard name ('stdstfl.dat' for variables and 'stdstolum.dat' for luminosity both in exec directory) is used. The file is written in ascii format but do not try to edit it. No protection against wrong formats.

When STORE is called, all the variables are written in a file. At the time of RESTORE there can be three kinds of variables (except for unchangeable ones): those already defined and appear in the file, already defined but do not appear in the file, and undefined variables. The first kind variables are overwritten. The second ones are kept (not eliminated) and the last ones are added.

When STORE LUMINOSITY is called, all the luminosity data at that time will be written in the specified file. When RESTORE LUMINOSITY is invoked, all the luminosity data in the present run is erased and replaced by the data in the file.

These two commands are introduced for convenience in splitting a job into two jobs for calculation and for output. For example, if you expect a long job but do not know what is to be printed/plotted. You write

```
WRITE  BEAM,  FILE='beam_file';
STORE;
STORE LUMINOSITY;
```

near the end of the long job. Then, you can print/plot the beam in the next job by

```
BEAM   FILE='beam_file';
RESTORE;
RESTORE LUMINOSITY;
PLOT  .........
```

Here, in the PLOT command you can use the variables you defined in the previous job. If you want different plots, you can repeat the second job.

However, keep in mind that these commands are not intended to split a job at arbitrary point. Only the user variables, luminosity data and particle data can be transfered to later jobs by STORE, RESTORE, and WRITE BEAM commands.

3.22 STOP

Stop **CAIN** run.

3.23 END

Indicates the end of input data. If absent, added at the end of file. At the beginning of **CAIN** run, the input file is read through until END (or end-of-file) and the command structure (command names and the terminator ";") is checked. Thus, the grammer beyond END is not checked in contrast to STOP.

4 Installation

The present version **CAIN**2.1e is working on a UNIX system. It should be very easy to install **CAIN**2.1e in other UNIX system.

To obtain **CAIN** by anonymous ftp

1. Go to the **CAIN** home page `http://www-acc-theory.kek.jp/members/cain/` and click at `cain21e.tar.gz`. Then you get `cain21e.tar.gz`. (about 375kB).[5]
2. '`gunzip`' it, you get `cain21e.tar`. It is a '`tar`'ed file. Move it to an appropriate directory on UNIX.
3. '`untar`' it by `tar -xvf cain21e.tar`
   Then, a new directory `cain21e` (overwritten if already exists) will be created under the current directory. This new directory contains five subdirectories `exec`, `in`, `out`, `source`, `doc`.[6] The directory `doc` contains a file `readme`.

You can also download the manual you are reading from the same home page (gzip'ed postscript file: manual-cain21e.ps.gz). [7]

4.1 Directory Structure

Everything is in a directory named `cain21e`. The directory structure is shown below.

```
cain21e ─┬source ─┬control ─┬main                   main program, reading input data
         │        │         ├deciph                 character manipulation to read input
         │        │         ├eval                   package to evaluate 'expressions'
         │        │         └basic                  basic physics routines
         │        ├physics ─┬bb      ─┬bbmain       entrance to bbfpack
         │        │         │         └bbfpack      subroutine package for beam field
         │        │         ├beamst                 beamstrahlung, coherent pair
         │        │         ├extfld                 external field
         │        │         ├lum                    luminosity integration
         │        │         ├laser   ─┬lsrmain      entrance to nllsr
         │        │         │         └nllsr        linear and nonlinear laser QED.
         │        │         └ppint                  incoherent particle-particle interaction
         │        ├local                            system dependent subroutines
         │        ├topdraw                          topdrawer
         │        ├util                             special functions, etc.
         │        ├util2                            histogram, scatter plot, etc.
         │        └include                          common blocks
         ├exec                                      load module, shell scripts
         ├in                                        input data
         ├out                                       outputs
         └doc                                       documents
```

[5] Or directly to the ftp site `ftp://lcdev.kek.jp/pub/Yokoya/cain21e.tar.gz`

[6] There may be one more directory `out0` containing the outputs from example data.

[7] Or from the ftp site `ftp://lcdev.kek.jp/pub/Yokoya/manual-cain21e.ps.gz`

4.2 System Dependent Subroutines

There are two subroutines which are system dependent.[8] They are all in the directory `cain21e/source/local`:

CLOCK1 returns the cpu time in seconds (from any origin) (for printing cputime only)

JOBDAT job date and time in 20 byte character string. (for printing header only)

Following is the source at KEK-HP work station. (`TIME` and `DATE` are KEK-HP system functions and `SECOND` is given by a C program.)

```
    SUBROUTINE CLOCK1(T)       SUBROUTINE JOBDAT(JOBTIM)
    IMPLICIT NONE              CHARACTER*20 JOBTIM
    REAL*8 T,SECOND            CALL TIME(JOBTIM(1:8))
    T=SECOND()                 JOBTIM(9:9)='('
    RETURN                     CALL DATE(JOBTIM(10:18))
    END                        JOBTIM(19:20)=') '
                               RETURN
                               END
```

4.3 Storage Requirements

A compressed array method has been used in **ABEL** but this was given up in **CAIN**, because it makes inter-lab collaboration in programming very hard. In **CAIN**, dimensions of large arrays are given by parameter statements. Major ones are the following. The given numbers are those in the present version. You can change them and re-compile all the files.

MXY in `include/bbcom.h`. For the mesh of beam-beam force. Upto MXY×MXY bins. $48\times\mathtt{MXY}^2 = 0.75$MB (MXY=128)

MMX,MMY in `bb/bbfpack/bbpkcm.inc`. Also for beam-beam force. Upto MMX×MMY bins. $40\times\mathtt{MMX}\times\mathtt{MMY} = 0.62$MB (MMX=MMY=128)
It is better to set MXY=MMX=MMY.

MMM in `include/lumcom.h`. For the mesh of luminosity. Upto $2^{\mathtt{MMM}} \times 2^{\mathtt{MMM}}$ bins. $152\times 2^{2\times\mathtt{MMM}} = 2.4$MB (MMM=7)

MWLUM in `include/lumcom.h`. Store differential luminosity. $8\times\mathtt{MWLUM} = 1.6$MB (MWLUM=200000).

MP in `include/beamcm.h`. Maximum number of macro particles, including photons, electrons, positrons, test particles, right-going and left-going. (Actual maximum number is 90% of MP because 10% is reserved for newly created particles in one time step.)
$192\times\mathtt{MP} = 19.2$MB (MP=100000)

[8] The older version used to have a system-dependent random number routine. It was replaced by a new one which is believed to be system independent. Please tell the programmer if you have any trouble in compiling `source/util/rand.f`.

MVPH in include/beamcm.h. Maximum number of virtual photons in a time step in an s-slice. $80\times$MVPH = 0.8MB (MVPH=10000)

MTSTP in include/tstpcm.h. Store the history of test particles. MTSTP is the maximum number of the number of time steps times the number of test particles.
$100\times$MTSTP = 0.5MB (MTSTP=5000).

The sum is about 30MB. The size of the load module is about 0.8MB.

4.4 Compilation

The directory cain21e/exec contains a csh script file @make which can be used when your system is UNIX, though the script might be system dependent. Otherwise, you have to write a compile command by yourself. @make works only when the current directory is cain21e/exec. (You can modify it so that it works anywhere. The only problem is that **CAIN** does not know in which directory you put him.)

When you compile all the source files, you say @make all, and when compiling only the files you changed, you should just say @make. (When @make all stopped due to a compilation error, @make will be enough next time, because @make all 'touches' all the files at the beginning.)

Before doing this, you have to do one thing: to specify the compiler option because it is quite system dependent. There is a csh envioronment variable foption in the file @make. It is passed to Makefiles in each directory. You need to change it for your system. Because of this, you cannot simply make in each directory without calling @make. Hopefully, an empty compiler option will do. (In an old version of **CAIN**, directories for include files were specified by a compiler option -I. Since this turned out to be system-dependent, the include statements in all the FORTRAN files in the present version, contains the directory names by relative paths. You must not therefore change the directory structure.)

4.5 Run

All the input data have to be written in the directory cain21e/in with file names having the extension '.i'. The file set sent to you has some example data (see the doc/readme file). The directory cain21e/exec contains a csh script file @go for execution. As @make it works only when the current directory is cain21e/exec

When you want to run **CAIN** with the input data example.i, for example, you would say @go example (without '.i'). If you use the same input file as in the previous run, @go suffices. TopDrawer output will be written on cain21e/out/example.tdr, OutFile on cain21e/out/example and OutFile2 on cain21e/out/example.out2.

If you want a submit job, please write an approproate shell script by yourself.

5 Physics and Numerical Methods

5.1 Coordinate

One of the basic assumptions of **CAIN** is that the main part (i.e., the part which contributes to the beam field dominantly) of the high energy beams consists of either (almost) right-going or left-going particles. The longitudinal coordinate s is the right-going direction. (The reason s is used instead of z is only historical since **ABEL**.) The x and y axes are perpendicular to s and (x, y, s) forms a right-handed orthonormal frame. The time coordinate t is always multiplied by the velocity of light.

In contrast to **ABEL**, **CAIN** does not use the longitudinal coordinate (z_1, z_2) attached to the beams.

5.2 Particle Variables

5.2.1 Arrays for Particles

All the particles (photons, electrons, positrons) carry the following variables.

`TXYS`(i) — $(i = 0, 1, 2, 3)$ Particle coordinates in meter.
Note that, in contrast to **ABEL**, the time and the s-coordinates are also defined for each particle. During tracking by `PUSH-ENDPUSH` command all the particles have basically the same time coordinate (an exception is the particles just created), whereas in some cases (e.g., after defined by `BEAM` command, after `DRIFT S=`s_1 command, etc.) they have different t but same s.
Also note that, in contrast to **ABEL**, s-coordinate does not simply change as $s_0 \pm ct$ but changes according to the instantaneous longitudinal velocity so that longitudinal mixing may occur for low energy or large angle particles.

`EP`(i) — $(i = 0, 1, 2, 3)$ Energy-momentum in units of (eV,eV/c).

`SPIN`(i) — $(i = 1, 2, 3)$ The polarization component (S_x, S_y, S_s) for electrons/positrons, and the Stokes parameter (ξ_1, ξ_2, ξ_3) for photons.
(S_x, S_y, S_s) is defined, as usual, in particle's rest frame. Therefore, it aquires the Thomas precession under Lorentz transformation by `LORENTZ` command.
For defining the Stokes parameter, one needs a set of orthonormal basis vectors $(\boldsymbol{e}^{(1)}, \boldsymbol{e}^{(2)}, \boldsymbol{e}^{(3)})$ with the third vector $\boldsymbol{e}^{(3)}$ parallel to the momentum. In **CAIN**, the first vector $\boldsymbol{e}^{(1)}$ is taken to be the unit vector along $\boldsymbol{e}_x - \boldsymbol{e}^{(3)}(\boldsymbol{e}_x \cdot \boldsymbol{e}^{(3)})$ and $\boldsymbol{e}^{(2)} = \boldsymbol{e}^{(3)} \times \boldsymbol{e}^{(1)}$. This is ill-defined when the momentum is exactly parallel to the x-axis but this possibility is simply ignored. Except for large angle photons, $(\boldsymbol{e}^{(1)}, \boldsymbol{e}^{(2)}, \boldsymbol{e}^{(3)})$ is almost equal to $(\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z)$ for right-going photons, and $(\boldsymbol{e}_x, -\boldsymbol{e}_y, -\boldsymbol{e}_z)$ for left-going photons.
See the next subsection for more detail on the polarization.

`GEN` — Generation. When a particle is generated by `BEAM` command by Twiss parameters, etc, `GEN`=1. Created particles such as beamstrahlung photons have `GEN` larger by one than that of the parent particle. (This is also true for the ‘spent’ parents.) GEN of the secondary particles due to particle-particle interaction (such as incoherent pairs) is the sum of GENs of the parents. In this case GENs of the parents do not change.

WGT Weight. Number of real particles represented by the macro-particle.

NAME 4-byte character string. Normally blanks. The test particles have Tnnn where nnn is a three-digit number. NAME of the particles created by incoherent (particle-particle) interactions starts with ‘I’. For example, ‘IBW ’, ‘IBH ’, ‘ILL ’ for the pairs created by incoherent Breit-Wheeler, Bethe-Heitler, Landau-Lifshitz processes, respectively.

5.2.2 Description of Polarization

Convention for electron/positron polarization

In most applications, one is interested in the helicity states. Therefore, one possible way of expressing the electron/positron spin is to store the information whether each macro-particle is in the helicity $h = +1$ state of -1 state. The unpolarized state is represented by an equal number of macro-particles with $h = +1$ and -1. The spin may flip at the interactions such as laser-Compton scattering and beamstrahlung.

However, this simple way cannot be applied to our case because, for example, a pure transverse polarization may become longitudinal during the precession in a magnetic field (beam-beam field or external field). In order to include such classical precession effects, the phase relation between the up and down components of the spinor is important.

This problem can be solved by using the density matrix. Let us express an electron(positron) state by a two-component spinor φ. The 2×2 density matrix $\rho^{(e)}$ is defined as

$$\rho^{(e)}_{ij} = \left\langle \varphi_i \varphi_j^\dagger \right\rangle, \qquad (i, j = 1, 2) \tag{1}$$

where † denotes the Hermitian conjugate and $\langle\ \rangle$ is the average over a particle ensemble. Since $\rho^{(e)}$ is Hermitian and its trace is unity by normalization, $\rho^{(e)}$ can be written as

$$\rho^{(e)} = \tfrac{1}{2}(1 + \boldsymbol{\zeta}\cdot\boldsymbol{\sigma}), \qquad \boldsymbol{\zeta} = \mathrm{Trace}(\rho^{(e)}\boldsymbol{\sigma}) = \left\langle \varphi\boldsymbol{\sigma}\varphi^\dagger \right\rangle \tag{2}$$

where $\boldsymbol{\sigma}$ is the Pauli matrices. The 3-vector $\boldsymbol{\zeta}$ is called polarization vector.

In the case of pure states, φ can be represented by a superposition of spin up(down) states $\varphi_\pm$:

$$\varphi = c_+\varphi_+ + c_-\varphi_-, \qquad |c_+|^2 + |c_-|^2 = 1. \tag{3}$$

With the standard representation of the Pauli matrices

$$\sigma_1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_2 = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma_3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \tag{4}$$

$\boldsymbol{\zeta}$ can be written as

$$\zeta_1 = 2\Re(c_+^* c_-), \quad \zeta_2 = 2\Im(c_+^* c_-), \quad \zeta_3 = |c_+|^2 - |c_-|^2, \tag{5}$$

and its length is unity: $|\boldsymbol{\zeta}| = 1$. **CAIN** allows $|\boldsymbol{\zeta}| \leq 1$ so that each macro-particle is in a mixed state, representing an ensemble of particles having almost the same energy-momentum and space-time coordinate.

If one observes the particle spin with the quantization axis $\boldsymbol{e}$ ($|\boldsymbol{e}| = 1$), the probability to be found in the spin $\pm\boldsymbol{e}$ state is given by $(1 \pm \boldsymbol{\zeta}\cdot\boldsymbol{e})/2$.

The polarization vector $\boldsymbol{\zeta}$ obeys the Thomas-BMT equation (33) in the absense of quantum phenomena.

Convention for photon polarization

A similar way is used for photon polarization, too. The polarization vector (3-vector) $\boldsymbol{\epsilon}$ (normalized as $|\boldsymbol{\epsilon}| = 1$) is orthogonal to the photon momentum $\boldsymbol{k}$. It can be represented by the components along two unit vectors $\boldsymbol{e}^{(1)}$ and $\boldsymbol{e}^{(2)}$ perpendicular to $\boldsymbol{k}$. The three vectors ($\boldsymbol{e}^{(1)}$, $\boldsymbol{e}^{(2)}$, $\boldsymbol{k}/|\boldsymbol{k}|$) form a right-handed orthonormal basis. The density matrix is defined as

$$\rho_{ij}^{(\gamma)} = \langle(\boldsymbol{\epsilon}\cdot\boldsymbol{e}_i)(\boldsymbol{\epsilon}^*\cdot\boldsymbol{e}_j)\rangle \,. \tag{6}$$

This is Hermitian with unit trace as in the case of electron density matrix so that it can be written as

$$\rho^{(\gamma)} = \tfrac{1}{2}(1 + \boldsymbol{\xi}\cdot\boldsymbol{\sigma}), \qquad \boldsymbol{\xi} = \mathrm{Trace}(\rho^{(\gamma)}\boldsymbol{\sigma}). \tag{7}$$

The 3-vector $\boldsymbol{\xi}$ is called the Stokes parameter. In the standard representation of the Pauli matrices, the three components of $\boldsymbol{\xi}$ have the meaning

- ξ_1 Linear polarization along the direction $(\boldsymbol{e}^{(1)} + \boldsymbol{e}^{(2)})/\sqrt{2}$ ($\xi_1 > 0$) or $(\boldsymbol{e}^{(1)} - \boldsymbol{e}^{(2)})/\sqrt{2}$ ($\xi_1 < 0$)
- ξ_2 Circular polarization
- ξ_3 Linear polarization along the direction $\boldsymbol{e}^{(1)}$ ($\xi_3 > 0$) or $\boldsymbol{e}^{(2)}$ ($\xi_3 < 0$).

The linear polarization can also be written as $\xi_3 = \xi_L \cos 2\phi_L$ and $\xi_1 = \xi_L \sin 2\phi_L$ ($\xi_L \geq 0$) where ξ_L is the magnitude of linear polarization and ϕ_L (modulo π) is the angle of the polarization plane measured from the $\boldsymbol{e}^{(1)}$-axis counterclockwise.

Completely polarized states have $|\boldsymbol{\xi}| = 1$. A single photon is always in a completely polarized state. Mixed states may have $|\boldsymbol{\xi}| < 1$.

In contrast to the case of electron/positron the polarization of a photon with a given momentum cannot be defined by the three numbers ξ_i: one has to define the $\boldsymbol{e}^{(1)}$-axis. The most general way is that every macro-photon carries its own $\boldsymbol{e}^{(1)}$-axis but this is too much redundant. **CAIN** adopts the convention that $\boldsymbol{e}^{(1)}$ is parallel to $\boldsymbol{e}_x - \boldsymbol{e}_x\cdot\boldsymbol{k}/|\boldsymbol{k}|$ where $\boldsymbol{k}$ is the photon momentum. This is ill-defined when $\boldsymbol{k}$ is parallel to $\boldsymbol{e}_x$ but it will not cause a serious problem. (For lasers $\boldsymbol{e}^{(1)}$-axis must be specified explicitly.)

Polarization-related processes

In any process involving polarizations, the transition rate (or crosssection) is given by multiplying the density matrices and by taking the trace. Therefore, the expressions for the rates are bilinear forms for each polarization vector, initial/final electron/positron or photon. The final polarization needs some comments. The transition rate is written in general as

$$W = \frac{1}{2}\int d\mathit{\Gamma}(w + \boldsymbol{g}\cdot\overline{\boldsymbol{\zeta}}) \tag{8}$$

where $\mathit{\Gamma}$ represents the final energy-momentum variables and w and $\boldsymbol{g}$ are functions of $\mathit{\Gamma}$. The vector $\overline{\boldsymbol{\zeta}}$ itself is not the final polarization. Its direction is defined by the setup of the detectors. What the term $\boldsymbol{g}\cdot\overline{\boldsymbol{\zeta}}$ means is that, if one observes the spin direction $\boldsymbol{e}$ ($|\boldsymbol{e}| = 1$), the probability to be found in the state $\pm\boldsymbol{e}$ is given by $\frac{1}{2}\int d\mathit{\Gamma}(w \pm \boldsymbol{g}\cdot\boldsymbol{e})$.

The final energy-momentum distribution is determined by $w(\mathit{\Gamma})$. For given $\mathit{\Gamma}$, the final polarization vector is (see [3], page 254)

$$\boldsymbol{\zeta} = \boldsymbol{g}(\mathit{\Gamma})/w(\mathit{\Gamma}). \tag{9}$$

Now, consider a process involving initial and final electrons, summing over other possible particles. The transition rate is written as

$$dW = \frac{1}{2}\int d\mathit{\Gamma}(w + \boldsymbol{f}\cdot\boldsymbol{\zeta}_i + \boldsymbol{g}\cdot\overline{\boldsymbol{\zeta}}_f + \overline{\boldsymbol{\zeta}}_f^T H\boldsymbol{\zeta}_i) \tag{10}$$

where the subscripts i and f denote initial and final variables, ^{T} represents transpose, and H is a 3×3 matrix. For given $\boldsymbol{\zeta}_i$, the final energy-momentum distribution is determined by $w + \boldsymbol{f}\cdot\boldsymbol{\zeta}_i$. In a Monte Carlo algorithm, $\mathit{\Gamma}$ is decided by using random numbers according to $w + \boldsymbol{f}\cdot\boldsymbol{\zeta}_i$. Once $\mathit{\Gamma}$ is decided, the final polarization is definitely (without using random numbers) given by

$$\boldsymbol{\zeta}_{f,\text{trans.}} = \frac{\boldsymbol{g}(\mathit{\Gamma}) + H(\mathit{\Gamma})\boldsymbol{\zeta}_i}{w(\mathit{\Gamma}) + \boldsymbol{f}(\mathit{\Gamma})\cdot\boldsymbol{\zeta}_i}. \tag{11}$$

This expression does not satisfy $|\boldsymbol{\zeta}| = 1$. If one does not allow a macro-particle in a mixed state, one has to choose a pure state by using random numbers.

The macro-particles which did not make transition must carefully be treated. One might say their final polarization is equal to $\boldsymbol{\zeta}_i$ but this is not correct because of the selection effect due to the term $\boldsymbol{f}\cdot\boldsymbol{\zeta}_i$.

The probability that a transion does not occur in a time interval Δt is $1-(\underline{w}+\underline{\boldsymbol{f}}\cdot\boldsymbol{\zeta}_i)\Delta t$, where the underlines indicates quantities integrated over the whole kinetic range of $\mathit{\Gamma}$. Consider an ensemble (one macro-particle) of N (real) particles having the polarization vector $\boldsymbol{\zeta}_{i,\alpha}$ ($\alpha = 1, 2, \ldots, N$). Each of these is a unit vector $|\boldsymbol{\zeta}_{i,\alpha}| = 1$ and the average over the ensemble is $\boldsymbol{\zeta}_i = \langle\boldsymbol{\zeta}_{i,\alpha}\rangle$.

Let us arbitrarily take the quatization axis $\boldsymbol{e}$. The probability in the state $\pm\boldsymbol{e}$ is $(1 \pm \boldsymbol{e}\cdot\boldsymbol{\zeta}_{i,\alpha})/2$ and the non-transition probability is $1 - (\underline{w} \pm \underline{\boldsymbol{f}}\cdot\boldsymbol{e})\Delta t$. Therefore, the sum of the final polarization along $\boldsymbol{e}$ over the ensemble is

$$\sum_\alpha\sum_\pm \frac{1 \pm \boldsymbol{e}\cdot\boldsymbol{\zeta}_{i,\alpha}}{2}[1-(\underline{w}\pm\underline{\boldsymbol{f}}\cdot\boldsymbol{e})\Delta t] = \sum_\alpha[\boldsymbol{e}\cdot\boldsymbol{\zeta}_{i,\alpha}(1-\underline{w}\Delta t)-\underline{\boldsymbol{f}}\cdot\boldsymbol{e}\Delta t] = N\boldsymbol{e}\cdot[\boldsymbol{\zeta}_i(1-\underline{w}\Delta t)-\underline{\boldsymbol{f}}\Delta t].$$

The axis $\boldsymbol{e}$ is arbitrary. Therefore, the sum of the final polarization vector is given by the above expression with $\boldsymbol{e}$ taken away. The total number of particles without transition is $N[1 - (\underline{w} + \underline{\boldsymbol{f}}\cdot\boldsymbol{\zeta}_i)\Delta t]$. Thus, the final polarization vector is

$$\boldsymbol{\zeta}_{f,\text{no trans.}} = \frac{\boldsymbol{\zeta}_i(1 - \underline{w}\Delta t) - \underline{\boldsymbol{f}}\Delta t}{1 - (\underline{w} + \underline{\boldsymbol{f}}\cdot\boldsymbol{\zeta}_i)\Delta t} \tag{12}$$

The average final polarization over the whole ensemble, with and without transition, is then given by

$$\begin{aligned}\boldsymbol{\zeta}_f &= [1-(\underline{w}+\underline{\boldsymbol{f}}\cdot\boldsymbol{\zeta}_i)\Delta t]\,\boldsymbol{\zeta}_{f,\mathrm{notrans.}}+\int d\mathit{\Gamma}[w(\mathit{\Gamma})+\boldsymbol{f}(\mathit{\Gamma})\cdot\boldsymbol{\zeta}_i]\Delta t\,\boldsymbol{\zeta}_{f,\mathrm{trans.}}\\ &= \boldsymbol{\zeta}_i+[\underline{\boldsymbol{g}}-\underline{\boldsymbol{f}}-\underline{w}\boldsymbol{\zeta}_i+\underline{H}\boldsymbol{\zeta}_i]\Delta t. \end{aligned} \tag{13}$$

If one can ignore the change of energy-momentum during the transition, the evolution of the polarization is described by the differential equation

$$\frac{d\boldsymbol{\zeta}}{dt}=\underline{\boldsymbol{g}}-\underline{\boldsymbol{f}}-\underline{w}\boldsymbol{\zeta}+\underline{H}\boldsymbol{\zeta}. \tag{14}$$

Polarization effects included in **CAIN**

The present version of **CAIN** does not include all the polarization effects. The following table shows what effects are included. In any case, the correlation of polarization between final particles is not taken into account.

		initial e$^\pm$	laser	final e$^\pm$	final γ
Beamstrahlung	e$^\pm$ →e$^\pm$ + γ	LT	–	LT	LT
Linear laser-Compton	e$^\pm$ + laser →e$^\pm$ + γ	LT	LT	LT	LT
Nonlinear laser-Compton	e$^\pm$ + n·laser →e$^\pm$ + γ	L	L*	L	L
		initial γ	laser	final e$^\pm$	
Coherent pair	γ →e$^+$+e$^-$	LT	–	LT	
Linear laser-Breit-Wheeler	γ + laser →e$^+$+e$^-$	LT	LT	LT	
Nonlinear laser-Breit-Wheeler	γ + n·laser →e$^+$+e$^-$	L	L*	L	
		initial	final pair		
Incoherent Breit-Wheeler	γ + γ →e$^+$+e$^-$	L	N		
Incoherent Bethe-Heitler	γ+e→e+e$^+$+e$^-$	N	N		
Incoherent Landau-Lifshitz	e+e→e+e+e$^+$+e$^-$	N	N		
		initial	final		
Bremsstrahlung	e+e→e+e+γ	N	N		

L Longitudinal spin of electron/positron (or circular polarization of photon).

T Transverse spin of electron/positron (or linear polarization of photon).

L* ±100% circular polarization only.

N Not computed. (No change for existing particles, zero for created particles)

– Irrelevant.

5.3 Beam Parameters

The `BEAM` command makes it possible to define a beam in terms of the conventional Twiss parameters. A beam is defined by many parameters described in Sec.3.3. Here, we will give formulas how to generate a beam using these parameters. An important point is that the beam is defined on a plane s=constant rather than t=constant. Thus, the longitudinal structure of the beam appears as the time structure. Note that t is larger at the bunch tail.

The parameters are

(t_0, x_0, y_0, s_0)	Reference point of the Twiss parameters. (m)
E_0	Reference energy. (eV)
$\beta_{x,y}, \alpha_{x,y}, \eta_{x,y}, \eta'_{x,y}$	Twiss parameters.
$\epsilon_{x,y}$	Geometric emittance. (rad·m)
σ_t	R.m.s. bunch length. (m)
σ_ε	R.m.s. relative energy spread.
$n_x, n_y, n_t, n_\varepsilon$	Gaussian tail cut off.
θ_x, θ_y	Orbit slope. (rad)
ψ_x, ψ_y	Crab angle. (rad)
ϕ_{xy}	Beam roll in the x-y plane. (rad)
$d\varepsilon/dt$	Coherent energy slope (1/m).

First, generate particle variables in usual accelerator coordinate:

$$\begin{aligned}
t_1 &= \sigma_t r_1 && (15)\\
\varepsilon_1 &= \sigma_\varepsilon r_2 + (d\varepsilon/dt)t_1 && (16)\\
x_1 &= \sqrt{2\epsilon_x u_1 \beta_x}\cos\varphi_1 + \eta_x \varepsilon_1 && (17)\\
x'_1 &= \sqrt{2\epsilon_x u_1/\beta_x}(-\alpha_x \cos\varphi_1 - \sin\varphi_1) + \eta'_x \varepsilon_1 && (18)\\
y_1 &= \sqrt{2\epsilon_y u_2 \beta_y}\cos\varphi_2 + \eta_y \varepsilon_1 && (19)\\
y'_1 &= \sqrt{2\epsilon_y u_2/\beta_y}(-\alpha_y \cos\varphi_2 - \sin\varphi_2) + \eta'_y \varepsilon_1 && (20)
\end{aligned}$$

where r_1 (r_2) is a Gaussian random number of zero mean and unit standard deviation cut at n_t (n_ε), u_j (j=1,2) is a random number of exponential distribution ($\propto e^{-u}$) cut at $u = n_x^2/2, n_y^2/2$ and φ_j a uniform random number in $(0, 2\pi)$.

These variables are then transformed to

$$\begin{aligned}
t &= t_0 + t_1 && (21)\\
\begin{pmatrix} x \\ y \end{pmatrix} &= \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + \begin{pmatrix} \psi_x \\ \psi_y \end{pmatrix} t_1 + \begin{pmatrix} \cos\phi_{xy} & -\sin\phi_{xy} \\ \sin\phi_{xy} & \cos\phi_{xy} \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} && (22)\\
\begin{pmatrix} x' \\ y' \end{pmatrix} &= \begin{pmatrix} \theta_x \\ \theta_y \end{pmatrix} + \begin{pmatrix} \cos\phi_{xy} & -\sin\phi_{xy} \\ \sin\phi_{xy} & \cos\phi_{xy} \end{pmatrix} \begin{pmatrix} x'_1 \\ y'_1 \end{pmatrix} && (23)\\
s &= s_0 && (24)
\end{aligned}$$

Finally, the energy-momentum is given by

$$\begin{aligned}
E &= E_0 + E_0 \varepsilon_1 && (25)\\
p_s &= \pm\sqrt{\frac{E^2 - m^2}{1 + x'^2 + y'^2}} && (26)\\
p_x &= |p_s|\, x' && (27)\\
p_y &= |p_s|\, y' && (28)
\end{aligned}$$

where ± is + for right-going beam and − for left-going (note that right/left appears only here) and m is the relevant particle mass in units of eV/c^2.

5.4 Solving Equation of Motion

Under the `PUSH` command, the equation of particle motion is solved step by step with the `Time` as the independent variable. The time step size is determined automatically for each particle. Smaller step size is used for low energy particles. On the other hand, an exact solution is used in the case of `DRIFT EXTERNAL` command which uses either the time or s as the independent variable.

5.4.1 Equation of motion under `DRIFT EXTERNAL` command

The present version of **CAIN** accepts a constant external field only. The covariant form of the equation of motion

$$\frac{dx^\mu}{d\tau} = \frac{1}{m} p^\mu, \qquad \frac{dp^\mu}{d\tau} = \frac{e}{m} F^\mu{}_\nu p^\nu. \tag{29}$$

where τ is the proper time and F^μ_ν the electromagnetic field tensor, can be solved exactly when the field is constant. The eigenvalues of the matrix $f^\mu_\nu \equiv eF^\mu_\nu/m$ is given by $\pm\omega_1$ and $\pm\omega_2$, where

$$\omega_1 = \sqrt{\tfrac{1}{2}(\sqrt{a^2+4b^2}+a)}, \qquad \omega_2 = i\sqrt{\tfrac{1}{2}(\sqrt{a^2+4b^2}-a)} \tag{30}$$

with

$$a = \frac{e^2}{m^2}(\boldsymbol{E}^2 - \boldsymbol{B}^2), \quad b = \frac{e^2}{m^2}(\boldsymbol{E}\cdot\boldsymbol{B}). \tag{31}$$

Then, the solution is

$$p^\mu(\tau) = \sum_{\omega=\omega_1,\omega_2} \frac{\pm 1}{\sqrt{a^2+4b^2}} \left[(\omega^2 + \tilde{f}^\mu{}_\alpha \tilde{f}^\alpha{}_\nu)\cosh\omega\tau + (\omega^2 f^\mu{}_\nu + b\tilde{f}^\mu{}_\nu)\frac{\sinh\omega\tau}{\omega}\right] p^\nu(0), \tag{32}$$

where the upper (lower) sign applies to ω_1 (ω_2) and $\tilde{f}^{\mu\nu} \equiv \frac{1}{2}\epsilon^{\mu\nu\alpha\beta} f_{\alpha\beta}$ with $\epsilon^{\mu\nu\alpha\beta}$ being the antisymmetric tensor of rank 4.

The classical spin motion of electrons is given by the Thomas-BMT equation

$$\frac{d\boldsymbol{S}}{dt} = -\frac{e}{m\gamma}\left[(\gamma a + 1)\boldsymbol{B}_T + (a+1)\boldsymbol{B}_L - \gamma(a + \frac{1}{\gamma{+}1})\beta \boldsymbol{e}_v \times \frac{\boldsymbol{E}}{c}\right] \times \boldsymbol{S}, \tag{33}$$

where a is the coefficient of anomalous magnetic moment and

$$\boldsymbol{B}_L = \boldsymbol{e}_v(\boldsymbol{B}\cdot\boldsymbol{e}_v), \qquad \boldsymbol{B}_T = \boldsymbol{B} - \boldsymbol{B}_L = \boldsymbol{e}_v \times (\boldsymbol{B}\times\boldsymbol{e}_v). \tag{34}$$

When the field is very strong, a is different from the well-known value $\alpha/2\pi + O(\alpha^2)$ but is a function of the field strength characterized the parameter Υ.

$$\Upsilon = \frac{e}{m^3}\sqrt{-(F^{\mu\nu}p_\nu)^2} = \frac{e}{m^3}\sqrt{(p^0\boldsymbol{E}_T + \boldsymbol{p}\times\boldsymbol{B})^2 + \boldsymbol{E}_L^2}. \tag{35}$$

The functional form of $a(\Upsilon)/a(0)$ is shown in Fig.2. Simple polynomial approximations are used in **CAIN**.

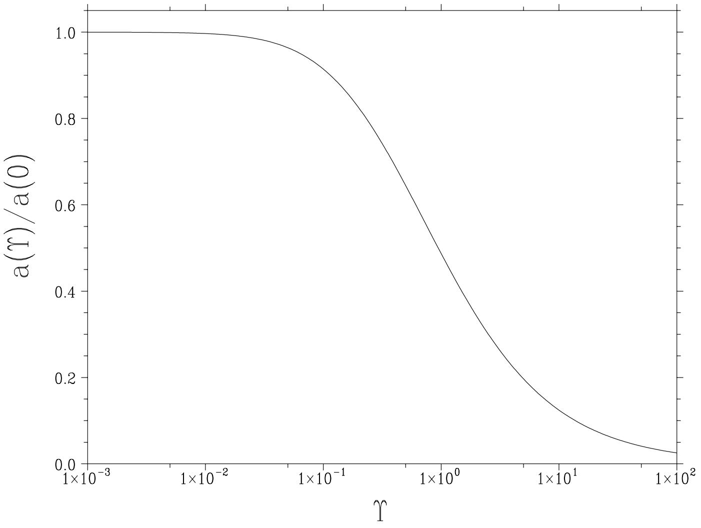


Figure 2: Field dependence of the anomalous magnetic moment of electron

5.4.2 Equation of motion under `PUSH` command

Solving the equation of motion in `PUSH` command is much more complicated because of the possible presense of the beam field. The equation of motion is in general written in the form

$$\frac{d\boldsymbol{r}}{dt} = \boldsymbol{v}(\boldsymbol{p}) = \frac{\boldsymbol{p}}{\sqrt{m^2+\boldsymbol{p}^2}} \tag{36}$$

$$\frac{d\boldsymbol{p}}{dt} = \boldsymbol{F}(\boldsymbol{r},\boldsymbol{p}) \tag{37}$$

The force $\boldsymbol{F}$ includes the beam field and the external field. The $\boldsymbol{p}$ dependence of $\boldsymbol{F}$ comes from $\boldsymbol{v}\times\boldsymbol{B}$ although very weak in the case of the beam field.

Given the initial variables $(\boldsymbol{r}_0, \boldsymbol{p}_0)$, a simple approximation after the time interval Δt is

$$\begin{aligned}
\boldsymbol{F}_0 &= \boldsymbol{F}(\boldsymbol{r}_0,\boldsymbol{p}_0) \\
\boldsymbol{p}_1 &= \boldsymbol{p}_0 + \boldsymbol{F}_0\Delta t \\
\boldsymbol{r}_1 &= \boldsymbol{r}_0 + \tfrac{1}{2}[\boldsymbol{v}(\boldsymbol{p}_0)+\boldsymbol{v}(\boldsymbol{p}_1)]\Delta t \\
\boldsymbol{F}_1 &= \boldsymbol{F}(\boldsymbol{r}_1,\boldsymbol{p}_1) \\
\boldsymbol{p} &= \boldsymbol{p}_0 + \tfrac{1}{2}(\boldsymbol{F}_0+\boldsymbol{F}_1)\Delta t \\
\boldsymbol{r} &= \boldsymbol{r}_0 + \tfrac{1}{2}[\boldsymbol{v}(\boldsymbol{p}_0)+\boldsymbol{v}(\boldsymbol{p})]\Delta t.
\end{aligned}$$

The error of $\boldsymbol{r}$ by these formulas is estimated by

$$\delta\boldsymbol{r} = \frac{1}{4}\frac{\boldsymbol{F}_1-\boldsymbol{F}_0}{\sqrt{m^2+\boldsymbol{p}_1^2}}(\Delta t)^2.$$

If this is not small enough, divide the interval Δt by an integer n_d. Note that $\delta\boldsymbol{r} \propto (\Delta t)^3$ because $\boldsymbol{F}_1 - \boldsymbol{F}_0$ is proportional to Δt. The total error, after multiplied by the number of intervals n_d, is proportional to $1/n_d^2$.

However, the above prescription is not really enough when there are extremely low energy particles (e.g., those from incoherent pair creation). It often happens that n_d so determined bocomes over several hundreds. In such a case the above error estimation may not be accurate at all.

When n_d is too large, **CAIN** tries the fourth-order Runge-Kutta integration. Starting from the whole interval Δt, it is divided by 2 at each step until the difference becomes small enough. This method is a little better than the simple formulas above but is still time consuming. So, the users should be aware that incoherent pair creation is expensive.

5.5 Luminosity

5.5.1 Luminosity Integration Algorithm

Let us denote the position-velocity distribution function of j-th beam (j=1,2) at time t by $n_j(\boldsymbol{r}, \boldsymbol{v}, t)$. It is normalized such that $\int \tilde{n}_j d\boldsymbol{r} d\boldsymbol{v}$ is the total number of particles in the j-th beam. The luminosity (per crossing) is in general given by

$$\mathcal{L} = \int \sqrt{(\boldsymbol{v}_1 - \boldsymbol{v}_2)^2 - (\boldsymbol{v}_1 \times \boldsymbol{v}_2)^2} \tilde{n}_1(\boldsymbol{r}, \boldsymbol{v}_1, t) \tilde{n}_2(\boldsymbol{r}, \boldsymbol{v}_2, t) d\boldsymbol{r} d\boldsymbol{v}_1 d\boldsymbol{v}_2 dt. \tag{38}$$

If all the particles in the j-th beam are ultrarelativistic and have almost the same velocity $\boldsymbol{v}_j$ ($|\boldsymbol{v}_j| \approx 1$), then the expression is simplified as

$$\mathcal{L} = (1 - \cos\phi) \int n_1(\boldsymbol{r}, t) n_2(\boldsymbol{r}, t) d\boldsymbol{r} dt \tag{39}$$

where ϕ is the polar angle between $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, and $n_j(\boldsymbol{r}, t)$ is the number density of the j-th beam. **CAIN** uses this formula with $\phi = \pi$, ignoring the velocity distribution and the crossing angle.

The integration is done by introducing the time step size Δt, longitudinal slice width Δ_s, transverse mesh size Δ_x and Δ_y. Summing the number of particles in each bin, the luminosity is given by

$$\mathcal{L} = C \sum_{i_x, i_y, i_s, i_t} N^{(1)}_{i_x, i_y, i_s, i_t} N^{(2)}_{i_x, i_y, i_s, i_t} \Delta_x \Delta_y \Delta_s \Delta_t \tag{40}$$

where C is an appropriate normalization factor, and $N^{(j)}_{i_x, i_y, i_s, i_t}$ is the number of particles of the beam j in the bin (i_x, i_y, i_s, i_t). A problem is how to determine the transverse size of the bin (Δ_t and Δ_s is mainly determined by the dynamics — they are actually specified by the user). If the bin is too large, detail of the distribution is lost, whereas if too small, statistical error becomes large because each bin will contain only a small number of macro-particles. **CAIN** adopts the following way.

At first, determine the size of the whole transverse region (w_x, w_y) such that most particles are contained there. Then, divide this region into as many bins $n \times n$ as allowed by the storage requirement (n must be a power of 2. **CAIN** uses $n = 128$.), and count the number of particles in each bin for both beams $N^{(j)}_k$ ($k = 0, 1, 2 \ldots, n^2 - 1$).

Figure 3: Bin numbering for luminosity integration. Example with n=8 and the double-sized bin n=4.

42	43	46	47	58	59	62	63
40	41	44	45	56	57	60	61
34	35	38	39	50	51	54	55
32	33	36	37	48	49	52	53
10	11	14	15	26	27	30	31
8	9	12	13	24	25	28	29
2	3	6	7	18	19	22	23
0	1	4	5	16	17	20	21

10	11	14	15
8	9	12	13
2	3	6	7
0	1	4	5

If the number of macro-particles in any of the neighbouring 4 bins are less than some number N_{min} (**CAIN** adopts 5) for both beams, then sum these numbers and put the sum into a larger bin $(2\Delta_x, 2\Delta_y)$. (For the example in Fig.3, the sum of the bins 12, 13, 14, and 15 in the figure on the left corresponds to the bin 3 on the right.) Otherwise, add $N_k^{(1)} N_k^{(2)}$ into the luminosity sum. This doubling of the bin size is repeated so long as $N^{(j)} < N_{min}$. In order to make this algorithm efficient, the bin numbering system is a little complicated. Instead of using two indices (i_x, i_y), the bins are numbered as in Fig.3. With this numbering, the sum of neighbouring bins can be simply written as

$$N_{4k}^n + N_{4k+1}^n + N_{4k+2}^n + N_{4k+3}^n = N_k^{n/2} \tag{41}$$

where N_k^n is the number of particles in the k-th bin ($k = 0, 1, 2, \ldots, n^2 - 1$) in $n \times n$ bin system.

5.5.2 Polarization

In the present version of **CAIN**, the particles are either photon or electron or positron. They all have two polarization eigenstates and can be specified by three real numbers, the Stokes parameter (ξ_1, ξ_2, ξ_3) for photons or the polarization vector $(\zeta_x, \zeta_y, \zeta_s)$ for electrons/positrons. Let us denote the three numbers in general by $\boldsymbol{s} \equiv (s_1, s_2, s_3)$.

Then, the crossection of a particular interaction integrated over a given final state (energy-momentum and polarization) is in general written in the form

$$\sigma = \sum_{i,j=0}^{3} \sigma_{i,j} s_i^{(R)} s_j^{(L)}, \tag{42}$$

where (R) and (L) represent the right and left-going particles and $s_0 = 1$ for notational convenience.

The number of events N during a beam collision is obtained by integrating eq.(42) with an appropriate factor over the momentum, the interaction volume and time:

$$N = \int d\boldsymbol{r}\, dt\, d\boldsymbol{p}^{(\boldsymbol{R})}\, d\boldsymbol{p}^{(\boldsymbol{L})} \sigma f \tag{43}$$

where f is the particle density functions with kinematic factors and is found in eq.(38). Since $\boldsymbol{s}^{(R)}$ and $\boldsymbol{s}^{(L)}$ depend on the particles in general, we get different coefficients from term to term of eq.(42). Thus, the number of events is written in the form

$$N = \sum_{i,j=0}^{3} \sigma_{i,j}\mathcal{L}_{i,j}, \qquad \mathcal{L}_{i,j} = \int d\boldsymbol{r}\, dt\, d\boldsymbol{p}^{(\boldsymbol{R})}\, d\boldsymbol{p}^{(\boldsymbol{L})} s_i^{(R)} s_j^{(L)} f \tag{44}$$

When the right and left-going beams are primary beams, the polarization is often uniform (*i.e.*, independent of energy-momentum and space-time) to a good approximation. In such a case $\mathcal{L}_{i,j}$ is simply given by $\mathcal{L}s_i^{(R)}s_j^{(L)}$, where $\mathcal{L} = \mathcal{L}_{00}$ is the total luminosity and $s_i^{(R,L)}$ is the polarization vectors of the beams. Then, the number of events is $\mathcal{L}\sigma$ where σ is given by eq.(44) with beam polarization value $s_i^{(R,L)}$ plugged-in.

Let us consider the helicity component in electron-electron collision. The helicity is approximately ζ_s and $-\zeta_s$ for right and left-going particles, respectively. The crosssections for four possible helicity combinations are

$$\begin{array}{rcl} \sigma_{++} & = & \sigma_{00} + \sigma_{30} - \sigma_{03} - \sigma_{33} \\ \sigma_{-+} & = & \sigma_{00} - \sigma_{30} - \sigma_{03} + \sigma_{33} \\ \sigma_{+-} & = & \sigma_{00} + \sigma_{30} + \sigma_{03} + \sigma_{33} \\ \sigma_{--} & = & \sigma_{00} - \sigma_{30} + \sigma_{03} - \sigma_{33} \end{array} \tag{45}$$

The number of events is written as

$$N = \sigma_{++}\mathcal{L}_{++} + \sigma_{-+}\mathcal{L}_{-+} + \sigma_{+-}\mathcal{L}_{+-} + \sigma_{--}\mathcal{L}_{--} \tag{46}$$

with

$$\begin{array}{rcl} \mathcal{L}_{++} & = & \frac{1}{4}(\mathcal{L}_{00} + \mathcal{L}_{30} - \mathcal{L}_{03} - \mathcal{L}_{33}) \\ \mathcal{L}_{-+} & = & \frac{1}{4}(\mathcal{L}_{00} - \mathcal{L}_{30} - \mathcal{L}_{03} + \mathcal{L}_{33}) \\ \mathcal{L}_{+-} & = & \frac{1}{4}(\mathcal{L}_{00} + \mathcal{L}_{30} + \mathcal{L}_{03} + \mathcal{L}_{33}) \\ \mathcal{L}_{--} & = & \frac{1}{4}(\mathcal{L}_{00} - \mathcal{L}_{30} + \mathcal{L}_{03} - \mathcal{L}_{33}) \end{array} \tag{47}$$

The total luminosity is $\mathcal{L}_{00}$. Note that the helicity is ξ_2 for photons. Thus, if both beams are photons, the above expression becomes

$$\begin{array}{rcl} \mathcal{L}_{++} & = & \frac{1}{4}(\mathcal{L}_{00} + \mathcal{L}_{20} + \mathcal{L}_{02} + \mathcal{L}_{22}) \\ \mathcal{L}_{-+} & = & \frac{1}{4}(\mathcal{L}_{00} - \mathcal{L}_{20} + \mathcal{L}_{02} - \mathcal{L}_{22}) \\ \mathcal{L}_{+-} & = & \frac{1}{4}(\mathcal{L}_{00} + \mathcal{L}_{20} - \mathcal{L}_{02} - \mathcal{L}_{22}) \\ \mathcal{L}_{--} & = & \frac{1}{4}(\mathcal{L}_{00} + \mathcal{L}_{20} + \mathcal{L}_{02} + \mathcal{L}_{22}) \end{array} \tag{48}$$

The helicity luminosity is calculated by `LUMINOSITY` command by specifying the operand `HELICITY`. If you want all 16 combinations or the linear polarization effects, you need to specify `ALLPOL` operand. For electrons, the expression (47) is not exact because the helicity is defined as $\boldsymbol{\zeta}\cdot\boldsymbol{p}/|\boldsymbol{p}|$ rather than $\pm\zeta_s$.

5.6 Beam Field

One of the basic assumptions of **CAIN** is that the most particles in the beams have high energy and are almost either right- or left-going. This assumption leads to the following facts:

- A field due to a particle is almost concentrated in a transverse plane with the same s-coordinate of the particle because of the Lorentz contraction.
- If the electric field is $\boldsymbol{E}$, the magnetic field is given by $\boldsymbol{B} = \pm c\boldsymbol{e}_s \times \boldsymbol{E}$, where $\boldsymbol{e}_s$ is the unit vector along the s-axis, c the velocity of light, and the upper (lower) sign is for the field created by the right(left)-goin beam.

In contrast to **ABEL**, **CAIN** does not assume that all the particles have the above property. Some particles may have low energies and large angles with respect to the s-axis. **CAIN** will work, unless the sum of their weight becomes a significant fraction of the beam. The equation of motion under the Lorentz force is integrated with possible low energies and large angles taken into account.

The calculation of the beam electric field is done in the following way. First, cut the right(left)-going particles into longitudinal slices (the width Δs is defined by the parameter `Smesh`). Within each slice the following Poisson equation is solved.

$$\Delta\Phi(x,y) = 2\pi\rho_Q(x,y), \qquad \boldsymbol{E} = \frac{2mr_e}{\Delta s}\nabla\Phi, \tag{49}$$

where m is the electron mass in units of eV/c^2, r_e the classical electron radius in meters, $\rho_Q(x,y)$ is the charge (divided by the elementary charge) per unit transverse area, then $\boldsymbol{E}$ is given in units of V/m.

For each slice and for each of right- and left going beams, a region $(x_c \pm w_x/2, y_c \pm w_y/2)$ is selected, where (x_c, y_c) is the center-of-mass and (w_x, w_y) is the width determined by the input parameters. The field created by the particles outside this region is ignored. Let us name this region [O].

[O] Fast Fourier Transformation

In the region [O], the Poisson equation is solved using the FFT. Eq.(49) can formally be solved as

$$\Phi(\boldsymbol{r}) = \int_{-\infty}^{\infty} G(\boldsymbol{r}-\boldsymbol{r}')\rho_Q(\boldsymbol{r}')d\boldsymbol{r}', \qquad G(\boldsymbol{r}) = \log|\boldsymbol{r}|\,. \tag{50}$$

Divide this region by $n_x \times n_y$ grid. Within each cell (i,j) $(i = 1, \dots n_x,\ j = 1, \dots n_y)$, the the density $\rho_Q(x,y)$ is approximated by $Q_{ij}/\Delta_x\Delta_y$, where $\Delta_x = w_x/n_x$, $\Delta_y = w_y/n_y$, and Q_{ij} is the total charge in the cell:

$$Q_{ij} = \int_{(x,y) \text{ in mesh } (ij)} \rho_Q(x,y)dxdy. \tag{51}$$

where (x_i, y_j) is the cell center coordinate. Then, eq.(50) becomes a sum over the cells.

$$\Phi(x_i, y_j) = \sum_{i',j'} G_{i-i',j-j'}Q_{i',j'}. \tag{52}$$

The kernel matrix $G_{i,j}$ has to be calculated by taking average over the source cell:

$$G_{i,j} = \frac{1}{\Delta_x\Delta_y}\int_{(x,y) \text{ in cell } (i,j)} \frac{1}{2}\log[(x_i - x)^2 + (y_j - y)^2]dxdy \tag{53}$$

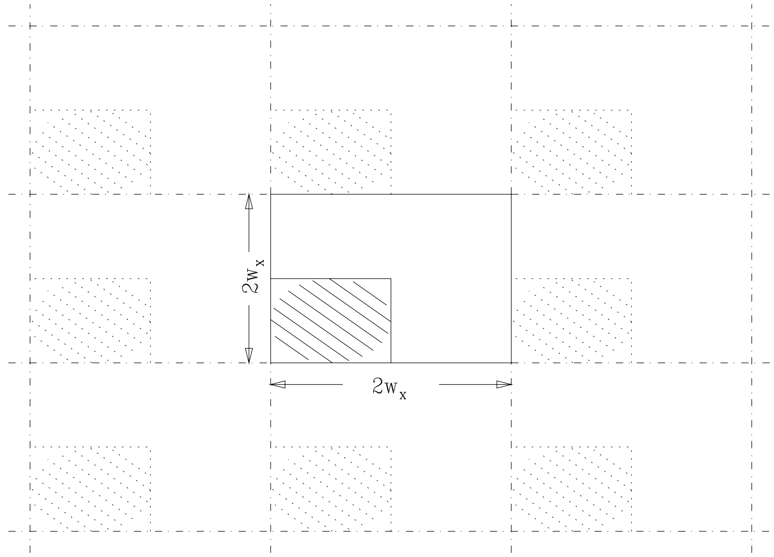


Figure 4: Doubled region for FFT. The solid frame indicates the doubled region for FFT and its left-bottom quadrant is the charge region $w_x \times w_y$. The region hatched by solid lines is the real charge region and that by dotted lines the ghost charge due to the periodicity of Fourier transformation.

This averaging is important when Δ_x/Δ_y is far from unity. The convolution in eq.(52) can be done efficiently by using FFT.

However, if we apply FFT for the finite region (w_x, w_y) instead of the infinite region in eq.(50), we would be assuming a periodic charge distribution, i.e., the charge distribution in (w_x, w_y) is infinitely repeated. To avoid this problem, we use the following trick. First double the region to $(2w_x, 2w_y)$ by padding zero in the extended region and carry out FFT. This still means a periodic charge distribution as depicted in Fig.4. However, if we use the kernel matrix with zero padded in the extended region ($G_{i,j} = 0$ if $n_x < i \leq 2n_x$ or $n_y < i \leq 2n_y$), the field due to the ghost charges will never reach the real charge region because their horizontal(vertical) distance is larger than w_x (w_y). Thus, the potential Φ in the region (w_x, w_y) is calculated correctly although incorrect in the extended region.

The obtained values of the potential are those at cell centers. They are interpolated by 2-dimensional cubic spline and differentiated to get $\partial\Phi/\partial x$ and $\partial\Phi/\partial y$.

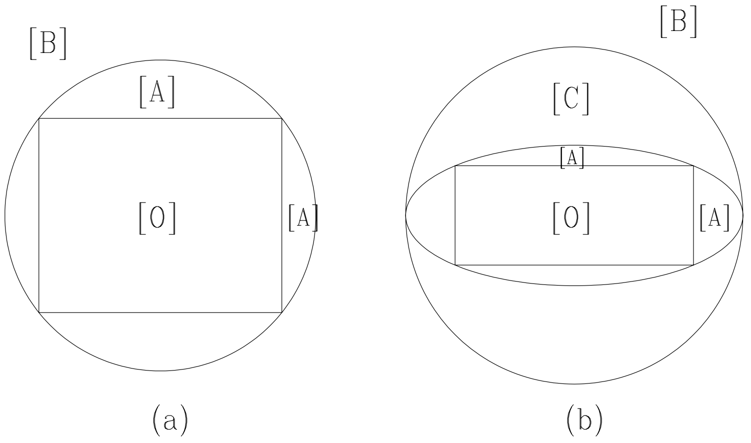
Outside the mesh region

When a charged particle gets out of the mesh region, the field created by it is ignored in **CAIN**2.1e. However, the force by the other beam is taken into account even if the particle is outside the mesh region of the other beam. To this end, **CAIN**2.1e adopts three methods, namely, [A] direct Coulomb force by the charge distribution in the mesh, [B] harmonic expansion in polar coordinate, and [C] harmonic expansion in elliptic coordinate.

Let (w_x, w_y) be the total width of the mesh region. If it is close to a square, or more precisely, if $0.8 < w_x/w_y < 1.25$, the whole region is divided into three regions [O],[A],[B], as depicted in Fig.5a. If the mesh region is far from square, the whole region is divided into four, [O],[A],[B],[C], as in Fig.5b. In the region [O] the mesh is used for calculating the field. In other regions, the methods mentioned above are used.

[A] Direct sum of Coulomb force

Figure 5: Regions for calculating the beam field



Since the sum is time consuming, this is used only in region [A], where two other methods fail to converge. The method is trivial and given by

$$\frac{\partial\Phi}{\partial x}+i\frac{\partial\Phi}{\partial y}=-\sum_{i,j}\frac{(x-x_i)+i(y-y_j)}{(x-x_i)^2+(y-y_j)^2}Q_{ij}, \tag{54}$$

However, this formula is not accurate when the bin size ratio Δ_x/Δ_y is far from unity. It is needed to take average over a bin when the bin is close to the field point (x, y). **CAIN** makes a table for the Coulomb force by a bin (Δ_x, Δ_y) for faster computation.

[B] Harmonic expansion in polar coordinate

In the region [B] the following formula is used.

$$\frac{\partial\Phi}{\partial x}-i\frac{\partial\Phi}{\partial y}=-\sum_{m=0}^{\infty}\left[\frac{r_0}{x+iy}\right]^{m+1}Q_m^B, \tag{55}$$

$$Q_m^B=\frac{1}{r_0}\int\rho_Q(x,y)\left[\frac{x+iy}{r_0}\right]^m dxdy. \tag{56}$$

Here, r_0 is arbitrary (introduced for avoiding overflow/underflow). The formula is valid for $x^2+y^2>r_{max}^2$, where $r_{max}=\sqrt{w_x^2+w_y^2}/2$ is the maximum radius of the mesh region.

[C] Harmonic expansion in elliptic coordinate

When $w_x > w_y$ (otherwise, exchange x and y), the elliptic coordinate (u, v) defined by

$$\begin{array}{ll} x=f\cosh u\cos v & \cosh(u+iv)=(x+iy)/f \\ y=f\sinh u\sin v & (u\geq 0,\quad 0\leq v<2\pi) \end{array} \tag{57}$$

is used. Here, f is chosen as

$$f = \sqrt{(w_x^2 - w_y^2)/2}. \tag{58}$$

The maximum of the radial-like coordinate u in the mesh region is

$$u_0 = \frac{1}{2}\log\frac{w_x + w_y}{w_x - w_y}, \tag{59}$$

which is taken at the four corners.

Then, the expansion of Φ is

$$\frac{\partial\Phi}{\partial x} - i\frac{\partial\Phi}{\partial y} = -\sum_{m=0}^{\infty} e^{-(m+1)(u-u_0+iv)} Q_m^C, \tag{60}$$

$$Q_m^C = \frac{2}{f}e^{-(m+1)u_0}\int \rho_Q(x,y)\frac{\sinh[(m+1)(u+iv)]}{\sinh(u+iv)}dxdy. \tag{61}$$

Actually, there is a finite relation between Q_m^B and Q_m^C:

$$Q_m^C = e^{-(m+1)u_0}\sum_{r=0}^{[m/2]}(-1)^r \left(\begin{array}{c} m-r \\ r \end{array}\right)\left(\frac{2r_0}{f}\right)^{m-2r+1} Q_{m-2r}^B. \tag{62}$$

The formula converges if $u > u_0$, which corresponds to the region [C] (and [B]) in Fig.5b. The truncation of the series is defined by the operand `NMOM` of the command `BBFIELD` (common to the two types of expansions for simplicity).

5.7 Laser

5.7.1 Laser Geometry

Define a coordinate system attached to a laser. Let $\boldsymbol{e}^{(3)}$ be the unit vector along the direction of propagation, and introduce a unit vector $\boldsymbol{e}^{(1)}$ perpendicular to $\boldsymbol{e}^{(3)}$ and another unit vector $\boldsymbol{e}^{(2)} = \boldsymbol{e}^{(3)}\times\boldsymbol{e}^{(1)}$. The three vectors $(\boldsymbol{e}^{(1)}, \boldsymbol{e}^{(2)}, \boldsymbol{e}^{(3)})$ form an orthonormal frame. Define the components of these vectors in the original frame $(\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_s)$ as

$$\boldsymbol{e}^{(1)} = \begin{pmatrix} V_{11} \\ V_{21} \\ V_{31} \end{pmatrix}, \qquad \boldsymbol{e}^{(2)} = \begin{pmatrix} V_{12} \\ V_{22} \\ V_{32} \end{pmatrix}, \qquad \boldsymbol{e}^{(3)} = \begin{pmatrix} V_{13} \\ V_{23} \\ V_{33} \end{pmatrix}. \tag{63}$$

Then, $V = \{V_{ij}\}$ is a 3×3 orthogonal matrix. Let (ξ, η, ζ) be the spatial coordinate in this frame. Define the origin of (u_1, u_2, u_3) as the laser focus and let (x_0, y_0, s_0) be its coordinate in the original frame and t_0 the time when the laser pulse center passes the origin. Introduce a time coordinate τ whose origin is t_0. Now, the relation between (t, x, y, s) and (τ, ξ, η, ζ) is

$$\tau = t - t_0 \tag{64}$$

$$\begin{pmatrix} \xi \\ \eta \\ \zeta \end{pmatrix} = \begin{pmatrix} V_{11} & V_{21} & V_{31} \\ V_{12} & V_{22} & V_{32} \\ V_{13} & V_{23} & V_{33} \end{pmatrix}\begin{pmatrix} x - x_0 \\ y - y_0 \\ s - s_0 \end{pmatrix}. \tag{65}$$

A plane wave is written in the form, in the (τ,ξ,η,ζ) coordinate,

$$\boldsymbol{E} = \Re\boldsymbol{E}_0 e^{ik(\boldsymbol{n}\cdot\boldsymbol{r}-\tau)} \tag{66}$$

where $k = 1/\lambdabar_L = 2\pi/\lambda_L$ is the wave number, $\boldsymbol{n}$ is the unit vector along the propagation direction of the wave component, and $\boldsymbol{E}_0$ is a complex vector perpendicular to $\boldsymbol{n}$. A laser beam is considered to be a superposition of plane waves with slightly different $\boldsymbol{n}$ and k. If the distribution of $\boldsymbol{n}$ around $\boldsymbol{e}^{(3)}$ and that of k are Gaussian and if one ignores the $\boldsymbol{n}$ dependence of $\boldsymbol{E}_0$, the laser field can be approximated by

$$\boldsymbol{E} = \Re\boldsymbol{E}_0 e^{ik(\zeta-\tau)}\sqrt{A}e^{i\Phi}, \tag{67}$$

where

$$A = \frac{1}{\sqrt{1+(\zeta/\beta_1)^2}}\frac{1}{\sqrt{1+(\zeta/\beta_2)^2}}\exp\left[-\frac{\xi^2}{\lambdabar\beta_1(1+(\zeta/\beta_1)^2)} - \frac{\eta^2}{\lambdabar\beta_2(1+(\zeta/\beta_2)^2)} - \frac{(\zeta-\tau)^2}{2\sigma_\tau^2}\right] \tag{68}$$

where β_i $(i=1,2)$ is the Rayleigh length and σ_τ is the r.m.s. pulse length.

The wave front is given by the contour of $k\zeta+\Phi$. If one defines $\boldsymbol{n}'$ by $k\boldsymbol{n}' = \nabla(k\zeta+\Phi)$, $\boldsymbol{n}'$ is nearly a unit vector and approximated by

$$\boldsymbol{n}' = \frac{\boldsymbol{e}^{(3)} + c_1\boldsymbol{e}^{(1)} + c_2\boldsymbol{e}^{(2)}}{\sqrt{1+c_1^2+c_2^2}}, \tag{69}$$

$$c_1 = \frac{\xi\zeta}{\beta_1^2+\zeta^2}, \qquad c_2 = \frac{\eta\zeta}{\beta_2^2+\zeta^2}. \tag{70}$$

In **CAIN**, when the relevant particle is at (t,x,y,s), or at (τ,ξ,η,ζ) in laser coordinate, the laser field is considered to be locally a plane wave with the power density $A(\tau,\xi,\eta,\zeta)P_{peak}$, wave number k, and the propagation direction $\boldsymbol{n}'(\xi,\eta,\zeta)$.

There is some problem on the polarization because eq.(67) does not exactly satisfy the Maxwell equation. For simplicity, the basis $(\boldsymbol{e}'^{(1)},\boldsymbol{e}'^{(2)},\boldsymbol{e}'^{(3)})$ $(\boldsymbol{e}'^{(3)}=\boldsymbol{n}')$ for polarization is defined in the following manner: $\boldsymbol{e}'^{(1)}$ is the unit vector along $\boldsymbol{e}^{(1)} - (\boldsymbol{e}^{(1)}\cdot\boldsymbol{n}')\boldsymbol{n}'$ and $\boldsymbol{e}'^{(2)} = \boldsymbol{n}'\times\boldsymbol{e}'^{(1)}$. (This is irrelevant if only the longitudinal polarization is needed.)

The Lorentz transformation is a little complicated because eq.(67) is far from a covariant form. The particle coordinates and the external fields are transformed immediately when `LORENTZ` command is invoked and the transformation parameters are forgotten. In the case of lasers, the transformation is not done immediately but instead the transformation parameters are stored. When the laser is called at every time step for each particle, the particle coordinates are Lorentz transformed back to the frame where the laser was defined, and the calculated parameters (A, ω', $\boldsymbol{e}'^{(1)},\boldsymbol{e}'^{(2)},\boldsymbol{e}'^{(3)}$) are transformed to the current Lorentz frame. Therefore, the Lorentz transformation is a little time-consuming.

5.7.2 Linear Compton Scattering

When the parameter `NPH=0` is specified in `LASERQED` command, the formulas of linear Compton scattering are used.

Let us define the following variables in the rest frame of the initial electron:

ω,ω'	Initial (laser) and final energies of the photon.
$\boldsymbol{k}$,$\boldsymbol{k}'$	Initial (laser) and final momenta of the photon.
θ, ϕ	Polar and azimuthal scattering angle of the photon.
$d\Omega$	Solid angle $= \sin\theta d\theta d\phi = (m/\omega'^2)d\omega' d\phi$.
$\boldsymbol{\xi}^{(L)}$,$\boldsymbol{\xi}'^{(L)}$	Photon Stokes parameters before and after collision as defined in[3](page 361).[9]

The range of ω' is given by

$$\frac{\omega}{1+\lambda} \le \omega' \le \omega, \qquad \lambda = \frac{2\omega}{m} \tag{71}$$

The Compton relation is

$$\frac{1}{\omega'} - \frac{1}{\omega} = \frac{1-\cos\theta}{m}. \tag{72}$$

The crosssection is given by eq(87.22) in [3]. [10]:

$$\frac{d\sigma}{d\Omega} = \frac{1}{8} r_e^2 \left(\frac{\omega'}{\omega}\right)^2 \times \Big[F_0 \,+\, F_3(\xi_3^{(L)} + \overline{\xi}_3^{\prime(L)}) + F_{11}\xi_1^{(L)}\overline{\xi}_1^{\prime(L)} + F_{22}\xi_2^{(L)}\overline{\xi}_2^{\prime(L)} + F_{33}\xi_3^{(L)}\overline{\xi}_3^{\prime(L)} \\ +(\boldsymbol{f}\cdot\boldsymbol{\zeta} + \boldsymbol{g}\cdot\overline{\boldsymbol{\zeta}}')\xi_2^{(L)} + (\boldsymbol{f}'\cdot\boldsymbol{\zeta} + \boldsymbol{g}'\cdot\overline{\boldsymbol{\zeta}}')\overline{\xi}_2^{\prime(L)} + \overline{\boldsymbol{\zeta}}'\cdot\mathcal{G}\cdot\boldsymbol{\zeta} + \ldots\Big] \tag{73}$$

See Sec.5.2.2 for the meaning of the bars on $\boldsymbol{\zeta}$ and $\boldsymbol{\xi}'^{(L)}$. The omitted terms are products of three and four among $\boldsymbol{\zeta}$, $\overline{\boldsymbol{\zeta}}'$, $\boldsymbol{\xi}^{(L)}$, and $\overline{\boldsymbol{\xi}}'^{(L)}$. (Actually, we need the terms $\zeta\times\xi\times\overline{\zeta}'$ and $\zeta\times\xi\times\overline{\xi}'$ but they are not found in literature.) The functions introduced in the above expression are:

$$F_0 = \frac{\omega}{\omega'} + \frac{\omega'}{\omega} - \sin^2\theta, \qquad F_3 = \sin^2\theta, \tag{74}$$

$$F_{11} = 2\cos\theta, \qquad F_{22} = \left(\frac{\omega}{\omega'} + \frac{\omega'}{\omega}\right)\cos\theta, \qquad F_{33} = 1 + \cos^2\theta, \tag{75}$$

$$\boldsymbol{f} = -\frac{1}{m}(1-\cos\theta)(\boldsymbol{k}\cos\theta + \boldsymbol{k}'), \qquad \boldsymbol{f}' = -\frac{1}{m}(1-\cos\theta)(\boldsymbol{k} + \boldsymbol{k}'\cos\theta), \tag{76}$$

$$\begin{bmatrix}\boldsymbol{g}\\ \boldsymbol{g}'\end{bmatrix} = \begin{bmatrix}\boldsymbol{f}\\ \boldsymbol{f}'\end{bmatrix} + \frac{\sin^2\theta}{m}\frac{\omega+\omega'}{\omega-\omega'+2m}(\boldsymbol{k}-\boldsymbol{k}'), \tag{77}$$

$$\begin{aligned}\mathcal{G} = \;& I(1+\cos^2\theta + \frac{\omega-\omega'}{2m}\sin^2\theta) + \frac{\omega+\omega'}{2m}(1+\cos\theta)(\boldsymbol{n}'\otimes\boldsymbol{n} - \boldsymbol{n}\otimes\boldsymbol{n}') \\ & -\frac{\omega-\omega'}{2m}[(\boldsymbol{n}+\boldsymbol{n}')\otimes(\boldsymbol{n}+\boldsymbol{n}') + (\boldsymbol{n}\times\boldsymbol{n}')\otimes(\boldsymbol{n}\times\boldsymbol{n}')] \\ & +\frac{\omega-\omega'}{m(\omega-\omega'+2m)}(1+\cos\theta)(\boldsymbol{k}-\boldsymbol{k}')\otimes(\boldsymbol{n}+\boldsymbol{n}') \\ & -\frac{\sin^2\theta + 2\cos\theta}{m(\omega-\omega'+2m)}(\boldsymbol{k}-\boldsymbol{k}')\otimes(\boldsymbol{k}-\boldsymbol{k}').\end{aligned} \tag{78}$$

[9] Actually, [3] adopts left-handed basis after collision so that the first and the second components of ξ' are $-\xi_1'$ and $-\xi_2'$ in [3]. Our $\xi'^{(L)}$ is not $(-\xi_1', -\xi_2', \xi_3')$ but (ξ_1', ξ_2', ξ_3').

[10] The term $\mathcal{G}$ is given in eq(4.6) in [4]

These formulas are used in their exact forms in **CAIN**.

Summation over the final polarization and the azimuthal angle ϕ gives the differential crosssection with respect to the final photon energy ω'. Introducing the variables $\overline{z}$ inplace of ω' by

$$\overline{z} = z/L_\lambda \qquad (0 \le \overline{z} \le 1), \qquad z = \log(\omega/\omega'), \qquad L_\lambda \equiv \log(1+\lambda), \tag{79}$$

$$\cos\theta = 1 - 2(e^z - 1)/\lambda, \tag{80}$$

we write the differential crossection as

$$\frac{d\sigma}{d\overline{z}} = 4\pi r_e^2 \frac{L_\lambda}{\lambda} F(\overline{z}), \tag{81}$$

where

$$F(\overline{z}) \;=\; \frac{1}{2L_\lambda}\left[1 + e^{-2z} - e^{-z}\sin^2\theta - h\cos\theta(1 - e^{-2z})\right], \tag{82}$$

$$h \;=\; \frac{\boldsymbol{k}}{\omega}\cdot\boldsymbol{\zeta}\,\xi_2^{(L)}. \tag{83}$$

Note that ξ_3 does not appear here because it is based on the scattering plane and, therefore, disappears after integration over the azimuthal angle. The function $F(\overline{z})$ satisfies $0 \le F \le 1$ for any $\overline{z}$ and λ and is $O(1)$ except when h is close to $+1$ and λ is extremely large. The Function $F(\overline{z})$ is plotted in Fig.6.

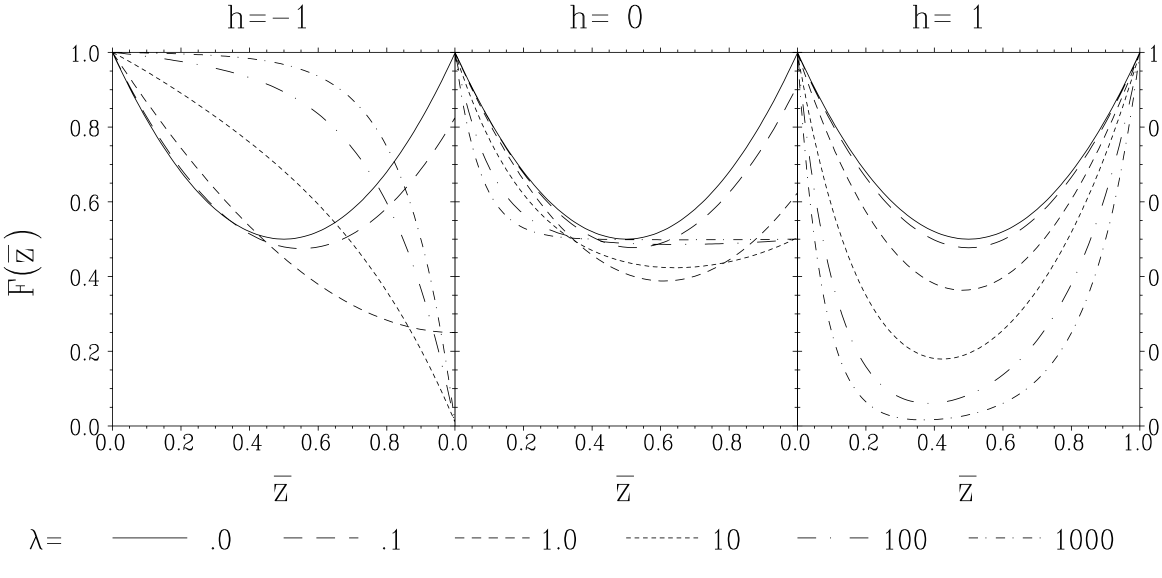


Figure 6: Function $F(\overline{z})$ for $h = 0, \pm 1$ for various values of λ

The total crosssection for given initial momenta and polarizations is given by

$$\sigma_{tot} = 4\pi r_e^2 \frac{L_\lambda}{\lambda} F_{int}(\lambda), \tag{84}$$

$$F_{int}(\lambda) = \frac{1}{2L_\lambda}\left\{\Bigl(1-\frac{4}{\lambda}-\frac{8}{\lambda^2}\Bigr)L_\lambda + \frac{1}{2}+\frac{8}{\lambda}-\frac{1}{2(1+\lambda)^2} + h\left[-\Bigl(1+\frac{2}{\lambda}\Bigr)L_\lambda + 2+\frac{1}{2(1+\lambda)^2}\right]\right\} \tag{85}$$

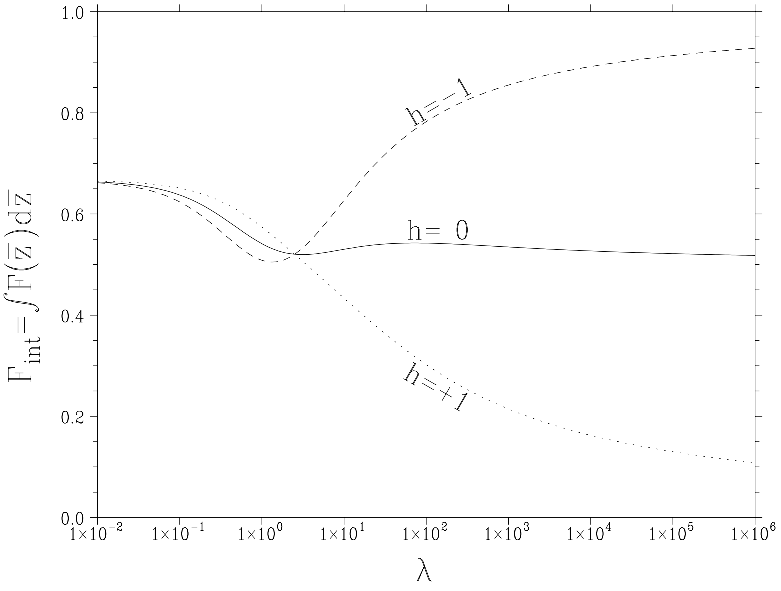


Figure 7: Function $F_{int}(\lambda)$ for $h = 0, \pm1$ as a function of λ. F_{int} is less than unity and is $O(1)$ unless h is close to $+1$ and λ is extremely large.

Let us driefly describe the algorithm of event generation.

1. Compute the total event rate P_0 in the given time interval using σ_{tot} without the factor F_{int}. Since $F_{int} \leq 1$, this is an over estimation of the rate. If P_0 is too large, divide the time interval by an integer N and repeat the following procedure N times.

2. Generate a random number r_1 uniform in (0,1). Reject if $r_1 \geq P_0$.

3. Compute F_{int} and multiply it to P_0. Reject if still $r_1 \geq P_0$. Otherwise accept. Note that the Lorentz transformation of $\boldsymbol{\xi}^{(L)}$ is not needed for the computation of h because ξ_2 is Lorentz invariant. Also note that input $\boldsymbol{\zeta}$ is defined already in the rest frame of electron. Only the Lorentz transformation of $\boldsymbol{k}$ is needed.

4. Generate two random numbers $\overline{z}$ and r_2 in (0,1). Repeat this step until $r_2 < F(\overline{z})$ is satisfied. Once or twice repetition is normally enough unless h is close to $+1$ and λ is very large.

5. Compute ω' from $\overline{z}$. Generate the azimuthal angle, compute the final polarization if needed, and go back to the laboratory frame. In this step many Lorentz transformations are needed.

5.7.3 Quantum Electrodynamics Involving a Strong Laser Field

When the laser field is strong, the simple formulas of Compton/Breit-Wheeler can nolonger be used. The laser strength is characterized by the parameter

$$\xi = \frac{\lambda\!\!\!^{-}_L}{m}\sqrt{\mu_0 c P} \tag{86}$$

where $\lambda\!\!\!^{-}_L$ is the laser wavelength $/(2\pi)$ in meter, m the electron rest mass in eV$/c^2$, c the velocity of light in m/s, $\mu_0 = 4\pi \times 10^{-7}$, and P the power density in Watt/m^2. When $\xi \ll 1$, the simple formas are enough but as ξ becomes large, the probability of absorbing more than one photon in the laser field cannot be ignored. When $\xi \gg 1$, the constant-field approximation[11] becomes good. If $\xi < O(1)$, the expansion in terms of the number of absorbed photons, n, shows good convergence. The expansion takes a relatively simple form when the laser is circularly polarized by 100%. The present version accepts such case only.

In the case of the nonlinear Compton process $e + laser \to e + \gamma$, the number of emitted photons per unit time can be expanded in the form[12]

$$W = \frac{\alpha m^2 \xi^2}{4E^2} \sum_{n=1}^{\infty} \int_0^{\omega_n} d\omega \left[(1 + h_e \overline{h}_{e'})F_{1n} + h_L(h_e + \overline{h}_{e'})F_{2n} + h_e \overline{h}_{e'} F_{5n} + \overline{h}_\gamma (h_L F_{3n} + h_e F_{4n})\right]. \tag{87}$$

E, ω Energy of the initial electron and final photon.

h_e, h_L Helicity of the initial electron and laser ($-1 \le h_e \le 1$, $h_L = \pm 1$).

$\overline{h}_{e'}$,$\overline{h}_\gamma$ 'Detector helicity' of the final electron and photon.

ω_n Maximum photon energy when n laser photons are absorbed:

$$\omega_n = \frac{n\lambda}{1 + \xi^2 + n\lambda} E. \tag{88}$$

λ Laser energy parameter: $\lambda = 4\omega_L E/m^2$. ($\omega_L$: laser photon energy)

F_{kn} Functions involving Bessel functions. See Sec.A.1 for the definition.

The photon emission angle is uniquely determined from n and ω.

The formula for the nonlinear Breit-Wheeler process $\gamma + laser \to e^+ + e^-$ can be written in a similar form. The total number of pair electrons per unit time summed over the positron polarization is

$$W = \frac{\alpha m^2 \xi^2}{2\omega^2} \sum_{n > (1+\xi^2)/\eta} \int_{E_n}^{\omega - E_n} dE \left[G_{1n} + h_L h_\gamma G_{3n} + \overline{h}_e (h_L G_{2n} + h_\gamma G_{4n})\right] \tag{89}$$

ω, E Energy of the initial photon and final electron.

h_γ, $\overline{h}_e$ Initial photon helicity and 'detector helicity' of the final electron.

η Laser energy parameter: $\eta = \omega_L \omega/m^2$ (ω_L: laser photon energy).

[11] This should be treated by the `CFQED` command. If users want, `EXTERNALFIELD` command will be rewritten so as to accept varying fields.

[12] The formulas bellow adopt the kinematics in a frame where the incident particle energy is much higher than electron mass and collide with the laser head-on. Actually, **CAIN** is valid in more general case. See Sec.A.

E_n Minimum energy of the final electron for given n:

$$E_n = \frac{1}{2}\left[1 - \sqrt{1-(1+\xi^2)/(n\eta)}\right]\omega. \tag{90}$$

G_{kn} Functions involving Bessel functions. See Sec.A.2 for the definition.

When `NPH`$\geq$ 1 is specified in `LASERQED` command, the above formulas are used with the terms upto n =`NPH`. See Sec.A for the algorithm of the event generation.

5.8 Beamstrahlung

5.8.1 Basic formulas

When the orbit of a high energy electron(positron) with energy $E_0 = mc^2\gamma$ is bent by a magnetic field B or by an electric field $\mathcal{E}$ with the curvature radius ρ, the critical energy of synchrotron radiation is given by

$$E_c = \overline{\Upsilon}E_0, \qquad \overline{\Upsilon} \equiv \frac{3}{2}\Upsilon, \quad \Upsilon = \gamma^2\frac{\lambda_C}{\rho} = \gamma\frac{B}{B_{Sch}} = \gamma\frac{\mathcal{E}}{\mathcal{E}_{Sch}} \tag{91}$$

where λ_C is the Compton wavelength, $B_{Sch} = m^2/e = 4.4\times 10^9$Tesla is Schwinger's critical field and $\mathcal{E}_{Sch} = cB_{Sch} = 1.32\times 10^{18}$V/m.

The energy spectrum of emitted photons is given by the Sokolov-Ternov formula. (Radiation angle is not included in **CAIN**. All the photons are emitted forward.) The number of photons per unit time in the interval $(x, x{+}dx)$ of the energy fraction $x = E_\gamma/E_0$ is

$$dW = \frac{\alpha m}{\sqrt{3}\pi\gamma}F_{00}dx, \qquad F_{00} = Ki_{5/3}(z) + \frac{x^2}{1-x}K_{2/3}(z). \tag{92}$$

Here, α is the fine structure constant and

$$z = \frac{E_\gamma}{E_c}\frac{1}{1-E_\gamma/E_0} = \frac{E_\gamma E_0}{E_c E'} = \frac{1}{\overline{\Upsilon}}\frac{E_\gamma}{E'} = \frac{1}{\overline{\Upsilon}}\frac{x}{1-x}. \tag{93}$$

$E' = E_0 - E_\gamma = E_0(1-x)$ is the final electron energy, K_ν the modified Bessel function and Ki_ν is its integral:

$$Ki_\nu(z) = \int_z^\infty K_\nu(z)dz. \tag{94}$$

The function F_00 is available as a **CAIN** function `FuncBS`, and so are some of the K_ν's and Ki_ν's. (See Sec.2.6).

5.8.2 Algorithm of event generation

The random number generation using the acception-rejection method is applicable when the distribution function is everywhere finite and is most efficient when the function is flat.

Since the function F_{00} is infinite at $E_\gamma \to 0$, the following variable y is introduced in **CAIN** instead of the photon energy fraction x in order to make the distribution function finite and relatively flat.[13]

$$x = \frac{\overline{\Upsilon} y^3}{1 - y^3 + \frac{1}{2}\overline{\Upsilon}(1+y^6)}, \qquad (0 < y < 1), \tag{95}$$

The number of photons during a time interval Δt in the photon energy range $(y, y+dy)$ is then given by

$$dn_\gamma = p_0 G(\Upsilon, y) dy, \tag{96}$$

where

$$p_0 = \frac{c_0}{\sqrt{3}\pi} \frac{1}{(1+\frac{1}{2}\overline{\Upsilon})^{1/3}} \frac{\alpha\gamma\Delta t}{\rho}, \qquad \left(\frac{\gamma\Delta t}{\rho} = \frac{\Delta t}{\lambda_C}\frac{\mathcal{E}}{\mathcal{E}_{Sch}}\right) \tag{97}$$

$$G(\Upsilon, y) = \frac{(1+\frac{1}{2}\overline{\Upsilon})^{1/3}}{c_0 \Upsilon} \frac{dx}{dy} F_{00}, \qquad c_0 = \frac{9}{2^{1/3}} \Gamma(2/3). \tag{98}$$

The function $G(\Upsilon, y)$ is less than or equal to unity for any Υ and y. It is plotted in Fig.8.

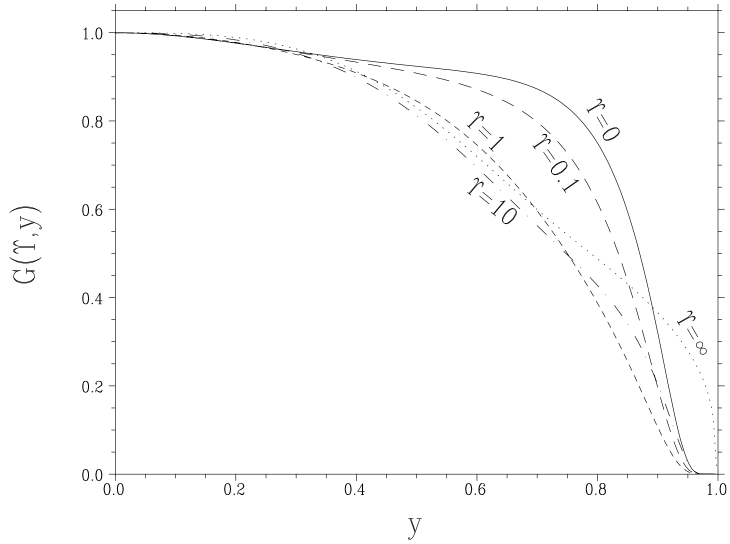


Figure 8: Function $G(\Upsilon, y)$ for various values of Υ. Unpolarized case only.

The photon generation in **CAIN** proceeds in the following way.

(1) Calculate p_0[14] for given field, electron energy, and time interval Δt.

[13] The definition of y has changed since **ABEL** and **CAIN**2.1b in order to keep high efficiency even when Υ is extremely large.

[14] This must be much smaller than 1. Otherwise, the probability of emitting more than one photon during Δt cannot be ignored. The maximum p_0 is defined by the keyword `PMAX`. When Υ is very large, p_0 is suppressed by the factor $(1+\frac{1}{2}\overline{\Upsilon})^{1/3}$. However, in defining Δt (by the number of steps in `PUSH` command) so as to make p_0 small enough, you have to omit this factor because the energy of some electrons can be much smaller after first radiation.

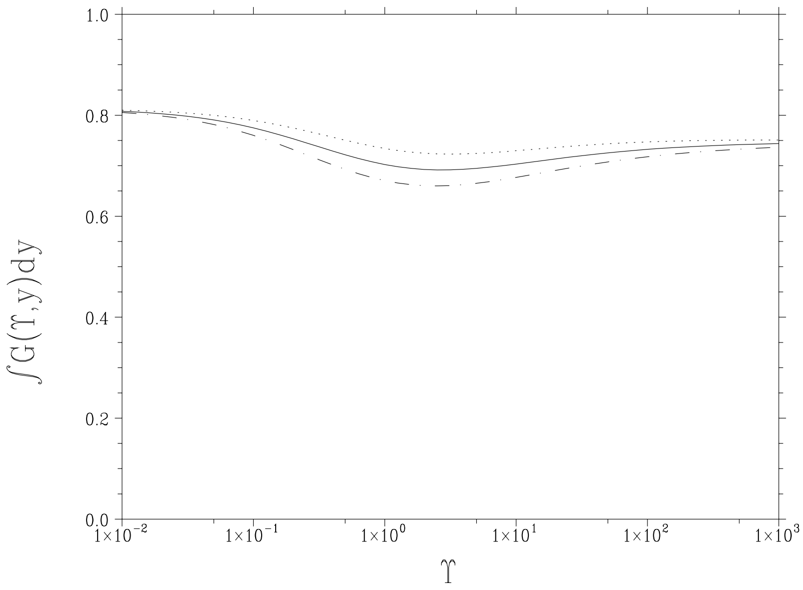


Figure 9: The acception probability in the step (6) as a function of Υ. The solid line is the unpolarized case The dot-dash and dotted lines are polarized cases with $\boldsymbol{\zeta}_i{\cdot}\boldsymbol{e}_2 = 1$ and -1, respectively.

(2) Generate one random number p which is uniform in (0,1).

(3) If $p \geq p_0$, reject emitting a photon. Otherwise,

(4) Generate one more random number y uniform in (0,1).

(5) Calculate $G(\Upsilon, y)$. (A polynomial approximation is used for $K_{2/3}$ and $Ki_{5/3}$. The relative error is less than 10^{-4}.)

(6) If $p \geq p_0 G(\Upsilon, y)$, reject emitting a photon. Otherwise,

(7) Emit a photon whose energy is given by eq.(95).

The cases when accepted in (3) but rejected in (6) cause a waste of time because the calculation of $G(\Upsilon, y)$ is the most time consuming. The probability to be accepted in (6) is plotted in Fig.9 is given by $\int_0^1 G(\Upsilon, y)dy$ and is plotted as a function of Υ. One finds the probability is very high for any Υ owing to the choice of the variable y.

5.8.3 Polarization

Polarization effects have been added since **CAIN**2.1. In order to describe polarizations, let us introduce an orthonormal basis vector ($\boldsymbol{e}_1$, $\boldsymbol{e}_2$, $\boldsymbol{e}_v$). Here, $\boldsymbol{e}_v$ is the unit vector along the initial electron velocity, $\boldsymbol{e}_1$ the unit vector along the direction of the transverse component of acceleration, and $\boldsymbol{e}_2 = \boldsymbol{e}_v \times \boldsymbol{e}_1$. If the acceleration is due to a transverse magnetic field, $-\boldsymbol{e}_2$ is the unit vector along the magnetic field (times the sign of charge). The transition rate from the initial electron polarization $\boldsymbol{\zeta}_i$ to the final polarization $\boldsymbol{\zeta}_f$ with the photon Stokes parameter $\boldsymbol{\xi}$ (based on the basis vector ($\boldsymbol{e}_1$, $\boldsymbol{e}_2$, $\boldsymbol{e}_v$)) is

$$\begin{aligned} dW \;=\; & \frac{\alpha dE_\gamma}{4\sqrt{3}\pi\gamma^2} \times \Bigg\{ F_{00}(1 + \boldsymbol{\zeta}_i{\cdot}\overline{\boldsymbol{\zeta}}_f) - xK_{1/3}(\boldsymbol{e}_2{\cdot}\boldsymbol{\zeta}_i) - \frac{x}{1-x}K_{1/3}(\boldsymbol{e}_2{\cdot}\overline{\boldsymbol{\zeta}}_f) \\ & -\frac{x^2}{1-x}\Big[(\boldsymbol{e}_v{\cdot}\boldsymbol{\zeta}_i)(\boldsymbol{e}_v{\cdot}\overline{\boldsymbol{\zeta}}_f)Ki_{1/3} + (\boldsymbol{\zeta}_i{\cdot}\overline{\boldsymbol{\zeta}}_f - \boldsymbol{e}_v{\cdot}\boldsymbol{\zeta}_i\boldsymbol{e}_v{\cdot}\overline{\boldsymbol{\zeta}}_f)K_{2/3}\Big] \end{aligned} \qquad (99)$$

$$+\frac{x}{1-x}K_{1/3}(\boldsymbol{e}_1\cdot\boldsymbol{\zeta}_i)\overline{\xi}_1-\frac{x(2-x)}{2(1-x)}K_{2/3}(\boldsymbol{e}_v\cdot\boldsymbol{\zeta}_i)\overline{\xi_2}+\left[K_{2/3}-\frac{x}{1-x}K_{1/3}(\boldsymbol{e}_2\cdot\boldsymbol{\zeta}_i)\right]\overline{\xi_3}\Bigg\}$$

where F_{00} is defined in eq.(92) and the argument of the Bessel functions is z. We omitted the terms involving $\boldsymbol{\zeta}_f$ and $\boldsymbol{\xi}$ simultaneously, which means to ignore the correlation of polarization between the final electron and photon. (See Sec.5.2.2 for the meaning of bars on $\boldsymbol{\zeta}_f$ and $\boldsymbol{\xi}$.)

The radiation energy spectrum summed over the final polarization is given by eq.(92) with F_{00} replaced by

$$F_0 = F_{00} - xK_{1/3}(\boldsymbol{e}_2\cdot\boldsymbol{\zeta}_i). \qquad (100)$$

Since the function $G(\Upsilon, y)$ with F_{00} replaced by F_0 has still the above mentioned property, the same algorithm of generating the photon energy can be used. ($G(\Upsilon, y)$ is slightly larger when $\boldsymbol{e}_2\cdot\boldsymbol{\zeta}_i = -1$ but still $G(\Upsilon, y) \leq 1$.)

For the given radiation energy $E_\gamma = xE_0$, the polarizations of the final electron and photon are calculated by the prescription described in Sec.5.2.2. Thus,

$$\boldsymbol{\zeta}_f = \frac{1}{F_0}\left\{F_{00}\boldsymbol{\zeta}_i - \frac{x}{1-x}K_{1/3}\boldsymbol{e}_2 - \frac{x^2}{1-x}\left[(\boldsymbol{e}_v\cdot\boldsymbol{\zeta}_i)\boldsymbol{e}_v Ki_{1/3} + [\boldsymbol{\zeta}_i - (\boldsymbol{e}_v\cdot\boldsymbol{\zeta}_i)\boldsymbol{e}_v]K_{2/3}\right]\right\}, \qquad (101)$$

$$\xi_1 = \frac{x}{1-x}\frac{K_{1/3}}{F_0}(\boldsymbol{e}_1\cdot\boldsymbol{\zeta}_i), \quad \xi_2 = -\frac{x(2-x)}{2(1-x)}\frac{K_{2/3}}{F_0}(\boldsymbol{e}_v\cdot\boldsymbol{\zeta}_i), \quad \xi_3 = \frac{K_{2/3} - \frac{x}{1-x}K_{1/3}(\boldsymbol{e}_2\cdot\boldsymbol{\zeta}_i)}{F_0}. \qquad (102)$$

In the case when the event generation is rejected, the polarization of the electron must be changed according to eq.(12):

$$\boldsymbol{\zeta}_f = \boldsymbol{\zeta}_i + [\boldsymbol{e}_2 - (\boldsymbol{e}_2\cdot\boldsymbol{\zeta}_i)\boldsymbol{\zeta}_i]\int_0^1 \frac{\alpha m}{\sqrt{3}\pi\gamma}xK_{1/3}(z)dx. \qquad (103)$$

In storage rings, the electron polarization builds up slowly along the direction of the magnetic field. This effect comes from the difference between the coefficient of $\boldsymbol{\zeta}_i$ and $\overline{\boldsymbol{\zeta}}_f$ (see eq.(14)):

$$\frac{x}{1-x}K_{1/3} - xK_{1/3} = \frac{x^2}{1-x}K_{1/3}. \qquad (104)$$

When Υ is small ($\lesssim 10^{-5}$ in storage rings), each term on the left hand side is proportional to Υ whereas the right hand side is Υ^2 because of cancellation. **CAIN** cannot reproduce such slow buildup, even if the computing time allows, because the approximate polynomials adopted do not have that accuracy. They are enough, however, for beam-beam problems.

5.8.4 Enhancement factor of the event rate

CAIN normally produces macro-photons such that the expected number of macro-photons per macro-electron is equal to the expected number of real photons per real electron. In some cases, however, too many macro-photons are created causing the memory overflow, or the statistics is too poor due to a small number of macro-photons. To solve this problem, a variable `WENHANCEMENT`$=w_{enh}$ is introduced in the `CFQED` command.

When $w_{enh} > 1$, more macro-photons are created. They have the weight smaller than that of the parent electron/positron by the factor $1/w_{enh}$. However, the recoil of electron/positron is taken into account only with the probability $1/w_{enh}$ so that their statistical property does not depend on w_{enh}.

When $w_{enh} < 1$, the event generation goes the same as in the case w_{enh}=1, but the final photons are stored in the memory only with the probability w_{enh}. The recoil of electron/positron is taken into acount regardless the photon is stored or not.

Thus, if there is no bug, w_{enh} does not cause any physical change.

5.9 Coherent Pair Creation

5.9.1 Basic formulas

When a high energy photon goes through a strong transverse field, it can create a real electron-positron pair. This process is known as 'coherent pair creation' and is characterized by the parameter

$$\chi = \frac{E_\gamma}{mc^2}\frac{B}{B_{Sch}} = \frac{E_\gamma}{mc^2}\frac{\mathcal{E}}{\mathcal{E}_{Sch}}, \tag{105}$$

where E_γ is the energy of the initial photon. (B_{Sch} and $\mathcal{E}_{Sch}$ are defined in eq.(91).) The probability of the process is exponentially small ($\propto e^{-8/(3\chi)}$) when χ is small.

Let us denote the energy and polarization vector of initial photon and final positron/electron by E_γ, $E_\pm$ ($E_+ + E_- = E_\gamma$), $\boldsymbol{\xi}$, and $\boldsymbol{\zeta}_\pm$. The transition rate is obtained by the following replacement in the formula (99):

$$\begin{aligned}
&E_\gamma \to -E_\gamma, \qquad E_0 \to -E_+, \qquad E' \to E_-, \qquad E_\gamma^2 dE_\gamma \to -E_+^2 dE_+, \\
&(\overline{\xi}_1, \overline{\xi}_2, \overline{\xi}_3) \to (\xi_1, -\xi_2, \xi_3), \\
&z = \frac{2}{3\Upsilon}\frac{E_\gamma}{E'} \to \eta = \frac{2}{3\chi}\frac{E_\gamma^2}{E_+E_-}.
\end{aligned} \tag{106}$$

Ignoring the terms related to the polarization correlation between the final electron and positron, we obtain

$$\begin{aligned}
dW \;=\; & \frac{\alpha m^2 dE_+}{4\sqrt{3}\pi E_\gamma^2} \times \left\{ F_{CP} - K_{2/3}\xi_3 + \frac{E_\gamma(E_+ - E_-)}{2E_+E_-} K_{2/3}\boldsymbol{e}_n\cdot(\overline{\boldsymbol{\zeta}}_+ - \overline{\boldsymbol{\zeta}}_-)\xi_2 \right. \\
& \left. + K_{1/3}\boldsymbol{e}_2\cdot\left(\frac{E_\gamma}{E_+}\overline{\boldsymbol{\zeta}}_+ + \frac{E_\gamma}{E_-}\overline{\boldsymbol{\zeta}}_-\right) + K_{1/3}(\boldsymbol{e}_1\xi_1 + \boldsymbol{e}_2\xi_3)\cdot\left(\frac{E_\gamma}{E_-}\overline{\boldsymbol{\zeta}}_+ + \frac{E_\gamma}{E_+}\overline{\boldsymbol{\zeta}}_-\right)\right\},
\end{aligned} \tag{107}$$

where F_{CP} is defined as

$$F_{CP} = Ki_{1/3} + \left(\frac{E_-}{E_+} + \frac{E_+}{E_-}\right)K_{2/3}. \tag{108}$$

K_ν is the modified Bessel function and Ki_ν is defined in eq.(94). Their arguments are η defined in eq.(106).

The transition rate summed over the final polarization is

$$dW = \frac{\alpha m^2 dE_+}{\sqrt{3}\pi E_\gamma^2}(F_{CP} - K_{2/3}\xi_3). \tag{109}$$

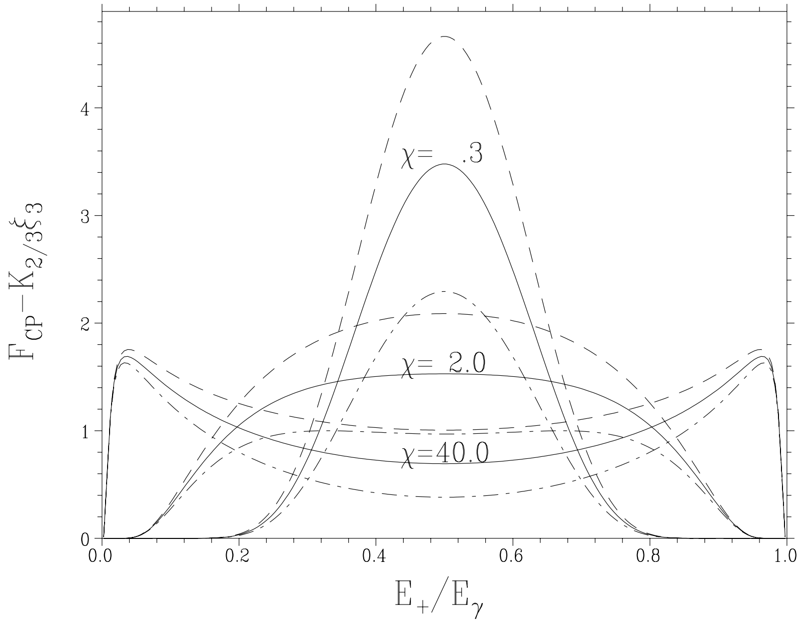


Figure 10: Function $F_{CP}-K_{2/3}\xi_3$ for three values of χ. The solid, dot-dash and dashed curves are for $\xi_3 = 0,\ 1,\ -1$, respectively. The curves for $\xi_3 = 0$ are normalized such that $\int F_{CP} dE_+/E_\gamma = 1$, and those for $\xi_3 \neq 0$ are drawn with the same scale as the corresponding $\xi_3 = 0$ curves.

The function $F_{CP} - K_{2/3}\xi_3$ is plotted in Fig.10 as a function of E_+/E_γ. The function F_{CP} is available as a **CAIN** function `FuncCP`, and its integral over $0 < E_+/E_\gamma < 1$ as `IntFCP`. (See Sec.2.6).

5.9.2 Algorithm of event generation

The total rate for given $\boldsymbol{\xi}$ is approximately

$$W \approx W_{app} \equiv \frac{\alpha m^2}{E_\gamma} U, \qquad U(\chi,\xi_3) \equiv e^{-\eta_0}\left\{\frac{\chi}{[c_1^3+c_2^3\chi]^{1/3}} - \frac{\xi_3\chi}{[(3c_1)^3+(5c_2)^3\chi]^{1/3}}\right\}, \quad (110)$$

where

$$\eta_0 \equiv \frac{8}{3\chi}, \qquad c_1 = \frac{16\sqrt{2}}{3\sqrt{3}}, \qquad c_2 = \frac{7\pi}{5}\frac{2^{2/3}}{3^{2/3}}\frac{1}{[\Gamma(5/6)]^2}.$$

In order to make the spectrum function flatter, **CAIN** introduces the variable y ($-1 < y < 1$) instead of $x = E_+/E_\gamma$ ($0 < x < 1$):

$$x = \frac{1}{2}\left(1 + \operatorname{sgn} y \sqrt{\frac{y'}{\eta_0 + y'}}\right), \qquad y' = -\log\left\{1 - \frac{y^2}{[1+(1-y^2)/(2\sqrt{\eta_0})]^2}\right\}. \quad (111)$$

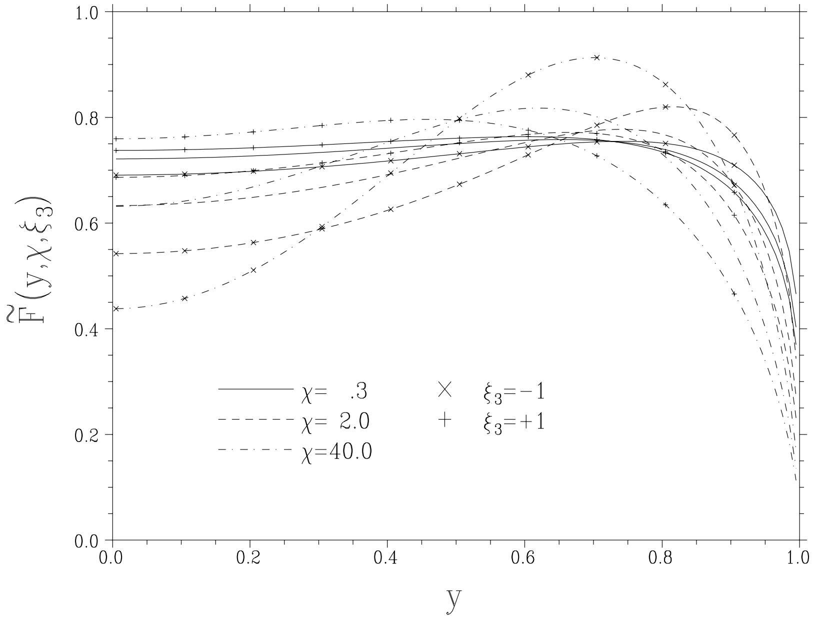
The spectrum function with respect to y then becomes

$$\frac{dW}{d(y/2)} = \frac{\alpha m^2}{E_\gamma} c_3 U(\chi,\xi_3)\tilde{F}(y,\chi,\xi_3), \qquad \tilde{F} \equiv \frac{(F_{CP}-K_{2/3}\xi_3)/\sqrt{3}\pi}{c_3 U}\frac{dx}{d(y/2)}, \quad (112)$$

where $y/2$ is used because the range of y is 2. The constant

$$c_3 = \frac{7\sqrt{\pi}}{6\Gamma(5/6)\Gamma(2/3)} = 1.35286\ldots.$$

Figure 11: Function $\tilde{F}(y,\chi,\xi_3)$ as a function of y for three values of χ =0.3, 2, 40 and for $\xi_3 = 0, \pm 1$. The parameter χ is indicated by the line mode and ξ_3 is by crosses (no cross for $\xi_3 = 0$).



is chosen so that $\tilde{F} \leq 1$ for any (y, χ, ξ_3). $\tilde{F}(y,\chi,\xi_3)$ is plotted in Fig.11.

Now, the event generation goes as

(1) Compute χ and reject if $\chi < 0.05$ (the rate is too small).

(2) Generate a uniform random number $0 < p < 1$ and compute $p_0 \equiv c_3 W_{app} \Delta t$.

(3) Reject if $p > p_0$.

(4) Generate another uniform random numbers $0 < q < 1$ and $-1 < y < 1$ and compute $\tilde{F}$. (Instead of q, one can also use p/p_0.)

(5) Reject if $q > \tilde{F}$.

(6) Accept and compute x from eq.(111) and $E_+ = xE_\gamma$, $E_- = (1-x)E_\gamma$.

(7) Compute the final polarization of $e^\pm$ from

$$(F_{CP} - K_{2/3}\xi_3)\boldsymbol{\zeta}_+ = \frac{1}{x} K_{1/3} \boldsymbol{e}_2 + \frac{1}{1-x} K_{1/3}(\xi_1 \boldsymbol{e}_1 + \xi_3 \boldsymbol{e}_2) + \frac{2x-1}{2x(1-x)} K_{2/3} \xi_2 \boldsymbol{e}_n \quad (113)$$

The formula for $\boldsymbol{\zeta}_-$ is obtained by replacing x by $1-x$.

(8) In the case of rejection, the polarization of the photon should be changed according to eq.(12):

$$\xi_{1,fin} = \xi_1 - a\xi_3\xi_1, \quad \xi_{2,fin} = \xi_2 - a\xi_3\xi_2, \quad \xi_{3,fin} = \xi_3 + a(1-\xi_3^2), \qquad (114)$$

$$a = \frac{\alpha m^2 \Delta t}{\sqrt{3}\pi E_\gamma} \int_0^1 K_{2/3}\left(\frac{2}{3\chi}\frac{1}{x(1-x)}\right) dx.$$

The error of the formula (110) does not cause any inaccuracy of event generation (but causes inefficiency).

5.10 Incoherent Pair Production

In addition to the interactions between a particle and a macro-scopic field such as beamstrahlung and laser-Compton, there are particle-particle interactions between e$^-$, e$^+$ and γ that have to be simulated. The present version of **CAIN** include the following processes:

Breit-Wheeler	$\gamma + \gamma \rightarrow \mathrm{e}^- + \mathrm{e}^+$
Bethe-Heitler	$\gamma + \mathrm{e}^\pm \rightarrow \mathrm{e}^\pm + \mathrm{e}^- + \mathrm{e}^+$
Landau-Lifshitz	$\mathrm{e} + \mathrm{e} \rightarrow \mathrm{e} + \mathrm{e} + \mathrm{e}^- + \mathrm{e}^+$
Bremsstrahlung	$\mathrm{e} + \mathrm{e} \rightarrow \mathrm{e} + \mathrm{e} + \gamma$

These QED processes have relatively large event rates even at high energies.

The treatment of these incoherent processes in **CAIN** is slightly different from other (coherent) processes since the event rates of the former are usually much smaller than the latter.

- The parent macro particles are not eliminated nor changed after interaction. The polarization change described in Sec.5.2.2 is not taken into account.

- More than one event may be generated in one time step because the change of parent partciles after the first event need not be taken into account.

In contrast to **ABEL**, **CAIN** does not assume the particles are ultra-relativistic. Therefore, unless the beam-beam field created by the pair particles becomes significant, their trajectories are correctly calculated. (Note, however, that only the beam field due to the on-coming beam is taken into account in the present version. The beam field in the same beam may have significant effects on low energy particles.)

Among the four QED processes above, the Breit-Wheeler process is treated as a fundamental process. Others are reduced to the former (or to the Compton process, in the case of Bremsstrahlung) by using the virtual photon approximation. As for the polarization, only the circular polarization of initial photons in the direct (non-virtual) Breit-Wheeler process is taken into account.

5.10.1 Breit-Wheeler Process

The differential crosssection with respect to the scattering angle of the final electron in the center-of-mass frame is given by

$$\frac{d\sigma_{BW}}{dc} = \frac{\pi}{2} r_e^2 \frac{m^2}{\omega^2} f, \qquad f = \frac{p}{\omega}(f_0 - f_2 h) \tag{115}$$

with

$$f_0 = \frac{\omega^2 + p^2c^2}{\omega^2 - p^2c^2} + \frac{1}{2}\left[1 - \left(1 - \frac{2m^2}{\omega^2 - p^2c^2}\right)^2\right] \tag{116}$$

$$f_2 = \frac{\omega^2 + p^2c^2}{\omega^2 - p^2c^2}\left(1 - \frac{2m^2}{\omega^2 - p^2c^2}\right) \tag{117}$$

where

ω, p Energy and momentum of final electron in the center-of-mass frame.

c Cosine of the scattering angle θ of the final electron in the center-of-mass frame.

h Product of circular polarizations of the two initial photons.

The total crosssection is

$$\sigma_{BW} = \pi r_e^2 \frac{m^2}{\omega^2} G \tag{118}$$

where

$$G = \int_0^1 f\, dc = 2\left(1 - h + \frac{2a^2-1}{2a^4}\right)\sinh^{-1} b + \frac{b}{a}\left[-(1+\frac{1}{a^2}) + 3h\right], \tag{119}$$

where

$$a = \omega/m, \qquad b = p/m, \qquad a^2 = b^2 + 1.$$

Events are generated by the following algorithm using inverse function.

(a) Compute ω, p, a, b, G and σ for given initial parameters (reject if $a \le 1$, *i.e.*, below threshold) and calculate the event probability for the given time step

$$P = \frac{w_1 w_2}{w}\frac{\sigma_{BW}\Delta t}{V}$$

where w_1 and w_2 are the weights of initial photons (number of real photons divided by that of macro photons), w the weight of the pair to be created, Δt the time interval and V the volume in which the macro phtons are located.

(b) If P is too large (say, > 0.1), divide the interval Δt (and P) by an integer n_{div}, and repeat the following procedure n_{div} times.

(c) Generate a random number $r_1 \in (0,1)$. Reject if $r_1 \ge P$.

(d) Generate another random number $r_2 \in (-1,1)$ and solve the equation

$$2\left(1 - h + \frac{2a^2-1}{2a^4}\right) z - (1-h)\tanh z + \frac{2a^2h - 1}{a^4}\sinh z \cosh z = |r_2|\, G. \tag{120}$$

with respect to z. Here z is defined by $|c| = |\cos\theta| = (a/b)\tanh z$ $(0 \le z \le \sinh^{-1} b)$. The left hand side is the integral of f from 0 to $|c|$. The sign of $c = \cos\theta$ is determined by the sign of r_2.

(e) Generate another random number $r_3 \in (0, 2\pi)$ and compute the transverse component of electron momentum by

$$\boldsymbol{p}_\perp = p\sin\theta(\boldsymbol{e}_1\cos\phi + \boldsymbol{e}_2\sin\phi) \tag{121}$$

where $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ are arbitrary unit vectors perpendicular to $\boldsymbol{e}_3$, the unit vector along the initial photon momentum in the center-of-mass frame. The latter is given by

$$\boldsymbol{e}_3 = \frac{(\omega+\omega_2)\boldsymbol{k}_1 - (\omega+\omega_1)\boldsymbol{k}_2}{\omega(2\omega+\omega_1+\omega_2)}$$

where $(\omega_j, \boldsymbol{k}_j)$ are the energy momentum of the photons in the original frame. The value of $\sin\theta$ should be computed from

$$\sin^2\theta = \frac{b^2 - \sinh^2 z}{b^2\cosh^2 z}, \qquad (\sin\theta \geq 0)$$

rather than from $|\cos\theta|$ because the latter is usually very close to unity when ω is much larger than the electron rest mass.

(f) Then, the momentum of the electron in the original frame is calculated by

$$\boldsymbol{p} = \frac{1}{2}\left[(1 + \frac{pc}{\omega})\boldsymbol{k}_1 + (1 - \frac{pc}{\omega})\boldsymbol{k}_2\right] + \boldsymbol{p}_\perp + \frac{\boldsymbol{k}\cdot\boldsymbol{p}_\perp}{2\omega + \omega_1 + \omega_2}\frac{\boldsymbol{k}}{2\omega} \tag{122}$$

where $\boldsymbol{k} = \boldsymbol{k}_1 + \boldsymbol{k}_2$. Note that $1 - p\,|c|\,/\omega$ must be computed from $e^{-z}/\cosh z$ in order to avoid round off errors.

(g) The momentum of positron is computed from the momentum conservation.

5.10.2 Virtual (almost real) photon approximation

To treat the Bethe-Heitler, Landau-Lifshitz and the Bremsstrahlung processes, the so-called almost-real-photon approximation, or equivalent photon approximation, or Weizäcker-Williams approximation, is employed. An electron is accompanied by virtual photons which look like real photons at ultra-relativistic limit. They interact with on-coming (real or virtual) photons incoherently. Thus, the Bethe-Heitler and Landau-Lifshitz processes above are reduced to the Breit-Wheeler process and the Bremsstrahlung to the Compton process:

Bethe-Heitler	$\gamma + `\gamma' \to \mathrm{e}^- + \mathrm{e}^+$
Landau-Lifshitz	$`\gamma' + `\gamma' \to \mathrm{e}^- + \mathrm{e}^+$
Bremsstrahlung	$\mathrm{e} + `\gamma' \to \mathrm{e} + \gamma$

where ‘γ’ is a virtual photon.

Let the electron energy be $E = m\gamma$ ($\gamma \gg 1$). The number of virtual photons with energy ω and transverse momentum $\boldsymbol{q}_\perp$ is given by

$$dn = \frac{\alpha}{\pi}\frac{d\omega}{\omega}\;\frac{1}{\pi}\frac{\boldsymbol{q}_\perp^2 d\boldsymbol{q}_\perp}{(\boldsymbol{q}_\perp^2 + \omega^2/\gamma^2)^2}, \qquad (|\boldsymbol{q}_\perp| \lesssim m) \tag{123}$$

where α is the fine structure constant. For given ω, the typical transverse momentum is very small, $q_\perp \sim \omega/\gamma$, so that it is not important in collision kinematics but, instead, the finite transverse extent $\sim 1/q_\perp$ can bring about significant effects. In the (transverse) configuration space, the above expression becomes

$$dn = \frac{\alpha}{\pi}\frac{d\omega}{\omega}\;K_1^2(\frac{\omega\rho}{\gamma})\frac{\omega}{\gamma^2}\frac{d\boldsymbol{r}_\perp}{\pi}, \qquad (\rho \gtrsim 1/m) \tag{124}$$

where $\boldsymbol{r}_\perp$ is the transverse coordinate with respect to the parent electron, $\rho = |\boldsymbol{r}_\perp|$, and K_1 the modified Bessel function.

The transverse momentum cut off $|\boldsymbol{q}_\perp| \lesssim m$ (or $\rho \gtrsim 1/m$) is somewhat umbiguous. It should depend on the momentum transfer of the whole process. This dependence is ignored in **CAIN** because the virtual photons are generated independently from the following processes and because it does not much affect the low energy pairs.

The lower limit ω_{min} of the integration over ω is, in our case, determined by the pair creation threshold. Let us introduce dimensionless variables $y = \omega/E$, $y_{min} = \omega_{min}/E$, and $x = \omega\rho/\gamma$. The total number of the virtual photons is given by[15]

$$n = \frac{\alpha}{\pi}\int_{y_{min}}^{1}\frac{dy}{y}\int_{y}^{\infty}K_1^2(x)2xdx \tag{125}$$

$$= \frac{\alpha}{\pi}\int_{y_{min}}^{1}\frac{dy}{y}V(y), \tag{126}$$

with

$$V(x) = x^2[K_0(x)K_2(x) - K_1^2(x)]. \tag{127}$$

When $y_{min} \ll 1$, the total number is

$$n = \frac{\alpha}{\pi}\left[\log^2 y_{min} - (2\log 2 - 2\gamma_{\scriptscriptstyle E} - 1)\log y_{min}\right], \tag{128}$$

where $\gamma_{\scriptscriptstyle E} = 0.577\ldots$ is Euler's constant. At very high energies the number of virtual photons per electron is $O(1)$, in spite of the small factor α/π, due to the factor $\log^2 y_{min}$.

5.10.3 Numerical methods

When Bethe-Heitler and/or Landau-Lifshitz processes are specified by PPINT command, **CAIN** generates virtual photons in each longitudinal slice at each time step and counts them in the same mesh as that generated by the LUMINOSITY command. The number of macro-virtual photons is somewhat arbitrary. In the present version it is determined such that the weight of the macro-virtual photons is equal to the maximum weight of the electrons in the on-coming beam (not equal to the weight of each parent electron in order to prevent low-weight electrons from generating many photons).

Since the y (energy) spectrum is approximately proportional to $\log y/y$ for small y, the spectrum becomes almost flat if one chooses $\log^2 y$ as the primary variable. To account for relatively large y too, **CAIN** adopts the variable η instead of y:

$$y = \exp\left[-\frac{\eta}{\sqrt{c+\eta}}\right], \qquad \eta = \frac{1}{2}\left(\log^2 y + \sqrt{\log^4 y + 4c\log^2 y}\right). \tag{129}$$

Here, $c > 0$ is introduced so that the function $G(\eta)$ defined later, is finite. It is chosen to be 0.2 but is almost arbitrary provided $c \gtrsim 0.1$. The maximum η is

$$\eta_{max} = \frac{1}{2}\left(\log^2 y_{min} + \sqrt{\log^4 y_{min} + 4c\log^2 y_{min}}\right). \tag{130}$$

[15] The upper limit $y = 1$ is not regorous. The recoil effect must be taken into account when y is large.

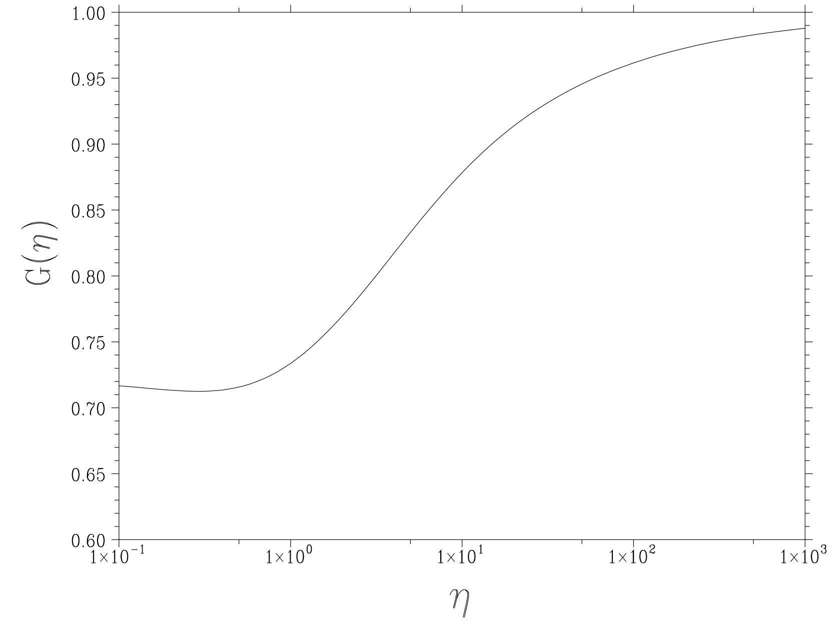


Figure 12: Function $G(\eta)$ defined in eq.(132). It is close to unity because only large η region is important. $G(0)$ is finite and depends on the parameter c. $G(0) < 1$ if $c > 0.1035\ldots$. Here, $c = 0.2$ is adopted.

Now, the spectrum with respect to η is

$$dn = \frac{\alpha}{\pi} G(\eta) d\eta, \tag{131}$$

with

$$G(\eta) = \frac{2c+\eta}{2(c+\eta)^{3/2}} V(y). \tag{132}$$

For $0 < \eta < \infty$, $G(\eta) \leq 1$ and close to 1 except for the small η region which is umimportant in practice. Thus,

$$n = \frac{\alpha}{\pi} \int_0^{\eta_{max}} G(\eta) d\eta \leq \frac{\alpha}{\pi} \eta_{max}. \tag{133}$$

For given η (or y) the distribution of x is proportional to $dV(x)/V(y)$ and, therefore, can be random-generated by using inverse function $V^{-1}(V)$.

The algorithm is as follows.

(a) From the given parameters, compute y_{min}, η_{max} and

$$P_0 \equiv \frac{\alpha}{\pi} \frac{\eta_{max}}{w}$$

where w is the weight of virtual photon to be created. P_0 is the expected number of macro-virtual photons. If P_0 is not small enough (say, > 0.1), divide it by an integer N and repeat the following steps N times.

(b) Generate a uniform random number $r_1 \in (0, 1)$. Reject if $r_1 \geq P_0$. Otherwise redefine r_1 by r_1/P_0.

(c) Generate a random number $r_2 \in (0, 1)$, define $\eta = r_2 \eta_{max}$ and calculate $G(\eta)$ from a table. Reject if $r_1 \geq G$. (The probability to be rejected here is small because G is close to unity.) Otherwise, accept.

(d) Calculate y using eq.(129) and $\omega = Ey$. If `LOCAL` option is specified, stop here and return $\boldsymbol{r}_\perp = (0,0)$. Otherwise, calculate the value of $V(y)$ from G using eq.(132).

(e) Generate a random number $r_3 \in (0,1)$ and solve the equation $r_3 = V(x)/V(y)$ with respect to x. This is done by using a table of inverse function of V.

(f) Compute $\rho = \lambda_e x/y$, λ_e being the Compton wave length.

(g) Generate a random number $r_4 \in (0,1)$ and compute the photon coordinate $\boldsymbol{r}_\perp = (\rho \cos 2\pi r_4, \rho \sin 2\pi r_4)$.

A Subroutine Package for Nonlinear Laser Interaction

For the nonlinear laser-electron and laser-gamma interactions, **CAIN** uses the subroutine package `nllsr`. This appendix is written for the description of the package. Some arguments of subroutines haven been modified when installed into **CAIN**, these are not reflected here. Also note that the notation of some variables is different.

Included items in `nllsr` are

- Compton scattering ($e + laser \rightarrow e + \gamma$) and Breit-Wheeler scattering ($\gamma + laser \rightarrow e^+ + e^-$) with nonlinear effects with respect to the laser intensity.
- The laser is circularly polarized. Longitidinal polarization of all the particles except that of the final electron of Compton scattering is included.

The laser intensity paramater ξ must not be too large because the multiple-photon expansion is used. When ξ is large, the computing time and the storage requirement will be enormous.

In both the Compton and Breit-Wheeler processes, the laser field intensity is characterized by the parameter

$$\xi = \frac{e\sqrt{-a{\cdot}a}}{m}, \tag{134}$$

where m the electron rest mass in eV$/c^2$, and a is the vector potential (4-vector) of the laser field. When the laser is circularly polarized completely, ξ is a constant and is given, in terms of the laser power density, as

$$\xi = \frac{\lambda\!\!\!^{-}_L}{m}\sqrt{\mu_0 c P} \tag{135}$$

where $\lambda\!\!\!^{-}_L$ is the laser wavelength $/(2\pi)$ in meter, c the velocity of light in m/s, $\mu_0 = 4\pi \times 10^{-7}$, and P the power density in Watt/m^2.

A.1 Compton Process

A.1.1 Formulas

In the following, $\approx$ is an approximation in the frame where the initial electron and laser collide head-on and the electron is ultra-relativistic.

p,k_L,p',k 4-momenta of initial electron, laser photon, final electron and emitted photon, respectively.

$\mathcal{E}$,ω_L,$\mathcal{E}'$,ω Energies of initial electron, laser photon, final electron and emitted photon, respectively.

λ Laser energy parameter: $\lambda = 2k_L{\cdot}p/m^2 \approx 4\omega_L\mathcal{E}/m^2$.

n Number of absorbed laser photon. The kinetic relation $q^\mu + nk_L^\mu = q'^\mu + k^\mu$ holds exactly. Here, q is defined as

$$q = p + \frac{m^2\xi^2}{2p{\cdot}k_L}k_L, \qquad (\ q^2 = (1+\xi^2)m^2\ ) \tag{136}$$

(p is replaced by p' for q') and is called quasimomentum.

x $x = (k{\cdot}k_l)/(p{\cdot}k_L) \approx \omega/\mathcal{E}$, $(0 < x < 1)$

v $v = x/(1-x)$, $x = v/(1+v)$. $(0 < v < \infty)$ $dv/(1+v)^2 = dx$.

v_n Maximum v for given n: $v_n = n\lambda/(1+\xi^2)$.

x_n Maximum x for given n: $x_n = v_n/(1+v_n) = n\lambda/(1+\xi^2+n\lambda)$.

h_L Laser helicity (-1 or $+1$)

h_e,$h_{e'}$ Initial and final electron helicities $(-1 \le h_e \le 1)$

h_γ Final photon helicity

$\overline{h}_{e'}$,$\overline{h}_\gamma$ 'Detector helicity' of the final particles. See section 65 of [3].

$\mathcal{E}_{eff}$ $= q^0 = \mathcal{E} + (\xi^2/\lambda)\omega_L$ is the effective energy of initial electron in the laser field.

θ_γ Final photon angle.

$$\theta_\gamma \approx \frac{m}{\mathcal{E}}\sqrt{(1+\xi^2)\left(\frac{v_n}{v}-1\right)} = \frac{m}{\mathcal{E}}\sqrt{n\lambda\frac{1-x}{x} - (1+\xi^2)}$$

z_n The argument of the Bessel functions in the following expressions:

$$z_n = 2n\frac{\xi}{\sqrt{1+\xi^2}}\sqrt{\frac{v}{v_n}\left(1-\frac{v}{v_n}\right)}$$

Number of photons per unit time is

$$W = \frac{\alpha m^2\xi^2}{4\mathcal{E}_{eff}}\sum_{n=1}^{\infty}\int_0^{x_n} dx[(1+h_e\overline{h}_{e'})F_{1n} + h_L(h_e+\overline{h}_{e'})F_{2n} + h_e\overline{h}_{e'}F_{5n} + \overline{h}_\gamma(h_LF_{3n}+h_eF_{4n})] \tag{137}$$

The terms involving h_γ and $h_{e'}$ simultaneously are ignored, i.e., the correlation of polarization between final particles is ignored. The ultra-relativistic approximation has been applied in the terms related to electron helicity (h_e and/or $h_{e'}$). (Note that the electron helicity is a Lorentz invariant quantity only in the ultra-relativistic limit.)

The sum over the final electron and photon helicities gives

$$W = \frac{\alpha m^2\xi^2}{\mathcal{E}_{eff}}\sum_{n=1}^{\infty}\int_0^{x_n} dx[F_{1n} + h_Lh_eF_{2n}] \tag{138}$$

The functions F_{kn} are defined by

$$F_{1n} = -\frac{1}{\xi^2}J_n^2 + \frac{1}{2}\left[1+\frac{v^2}{2(1+v)}\right](J_{n-1}^2 + J_{n+1}^2 - 2J_n^2) \tag{139}$$

$$F_{2n} = \left(\frac{1}{2}-\frac{v}{v_n}\right)\frac{v(1+v/2)}{1+v}(J_{n-1}^2 - J_{n+1}^2) \tag{140}$$

$$F_{3n} = \left(\frac{1}{2} - \frac{v}{v_n}\right)\left[1 + \frac{v^2}{2(1+v)}\right](J_{n-1}^2 - J_{n+1}^2) \tag{141}$$

$$F_{4n} = -\frac{v}{1+v}\frac{1}{\xi^2}J_n^2 + \frac{1}{2}\frac{v(1+v/2)}{1+v}(J_{n-1}^2 + J_{n+1}^2 - 2J_n^2) \tag{142}$$

$$F_{5n} = -\frac{v^2}{1+v}\frac{1}{\xi^2}J_n^2 \tag{143}$$

F_{1n}, F_{2n}, F_{3n}, F_{4n} are identical to D_{1n}, D_{2n}, N_{1n}, N_{1n} divided by $8\xi^2$ in Tsai's paper[5], although the expressions in his paper look much more complicated.

Once x and n are given, the final momentuma are given, in any frame, by

$$k^\mu = xp^\mu + \left[n(1-x) - \frac{2+\xi^2}{\lambda}x\right]k_L^\mu + mx\sqrt{n\lambda\frac{1-x}{x} - (1+\xi^2)}\,(e_1^\mu\cos\phi + e_2^\mu\sin\phi) \tag{144}$$

$$p'^\mu = p^\mu + \left(n - \frac{\xi^2}{\lambda}\frac{x}{1-x}\right)k_L^\mu - k^\mu. \tag{145}$$

Here, ϕ is the azimuthal angle in a head-on frame (therefore its distribution is uniform in [0,2π]) and e_2^μ and e_1^μ are given by

$$e_2^\mu = \left[\begin{array}{c} 0 \\ \frac{\boldsymbol{p}\times\boldsymbol{k}_L}{|\boldsymbol{p}\times\boldsymbol{k}_L|} \end{array}\right], \qquad e_1^\mu \equiv -\frac{\epsilon_{\mu\nu\alpha\beta}e_2^\nu p^\alpha k_L^\beta}{p\cdot k_L} = \left[\begin{array}{c} |\boldsymbol{p}\times\boldsymbol{k}_L|\,/p\cdot k_L \\ \boldsymbol{e}_2\times(\mathcal{E}\boldsymbol{k}_L - \boldsymbol{p}\omega_L)/p\cdot k_L \end{array}\right], \tag{146}$$

where $\epsilon^{\mu\nu\alpha\beta}$ is the completely anti-symmetric tensor ($\epsilon^{0123} = +1$). These vectors satisfy

$$p\cdot e_j = k_L\cdot e_j = 0, \quad e_j\cdot e_j = -1, \quad (j = 1, 2), \qquad e_1\cdot e_2 = 0. \tag{147}$$

The vector e_2^μ in eq.(146) is ill-defined when $\boldsymbol{p}$ and $\boldsymbol{k}_L$ are colinear in the original frame. In such a case the spatial part of e_2^μ is an arbitrary unit vector perpendicular to $\boldsymbol{k}_L$.

A.1.2 Usage

Initialization

First you must initialize the event generators, *i.e.*, create a table of the functions F_{1n} and F_{2n}. They are three-argument functions. In the program, ξ, λ, and $y = v/v_n$ $(0 \le y \le 1)$ are used as the independent variables. To initialize the generators, you must specify the range and number of mesh points for these variables.

The present version allows to enhance the probability of a part of spectrum. See below.

```
CALL NLCPST(MY,MPH,MXI,MLM,XIMAX,LMMAX,LENHCP,ENHCPF,IRTN)
```

MY	Number of y's. (i-th y point is $y_i = i/\mathtt{MY}$, $i = 0, \mathtt{MY}$)
MPH	Maximum number of laser photons. ($n = 0, \mathtt{MPH}$)
MXI	Number of ξ's. ($\xi_j^2 = \mathtt{XIMAX}^2 \times j/\mathtt{MXI}$, $j = 0, \mathtt{MXI}$)
MLM	Number of λ's. ($\lambda_l = \mathtt{LMMAX} \times l/\mathtt{MLM}$, $l = 0, \mathtt{MLM}$)

XIMAX	Maximum ξ.
LMMAX	Maximum λ.
LENHCP	Flag to apply a rate enhancement function.
ENHCPF	Enhancement function name declared external. Used when LENHCP$\geq$1.
IRTN	Return code.

A second call of NLCPST will replace the parameters and the arrays created in the first call.

The enhancement function, if needed, has to be defined as

```
FUNCTION ENHCPF(Y)
REAL*8 ENHCPF,Y
ENHCPF=......
RETURN
END
```

The enhancement function has to be a function of $y = v/v_n$ $(0 \leq y \leq 1)$ only and it must be ≥ 1 for all y. The probability functions F_{kn} are multiplied by ENHCPF(y). Note that y close to 1 represents events with high energy photons and, hence, low energy recoil electrons. See NLCPGN for how the enhancement should be treated in the simulation.
Possible errors

IRTN=1000	Memory insufficient. You have to reduce MY×MPH×MXI×MLM or increase the parameter MW in the FORTRAN source.
IRTN=1001 to 1004	Either one of MY,MPH,MXI,MLM is too large.
IRTN=1100	The enhancement function less than 1 at some y.

Event generation

After initialization, an event can be generated by

```
CALL NLCPGN(PE1,WL,NL,HE1,HL,PD,DT,PMAX,IRR,NPH,PE2,HE2,PG,HG,
            PROB,WGT,IRTN)
```

Input variables

PE1	Array of dimension (0:3). Initial electron 4-momentum (eV/c).
WL	Laser photon energy (eV).
NL	Array of dimension 3. Unit vector along the laser direction.
HE1	Initial electron helicity. $(-1 \leq$HE1$\leq 1)$.
HL	Laser helicity. (+1 or -1).
PD	Laser power density (Watt/m^2).
DT	Time interval times velocity of light. (meter).

PMAX Maximum radiation probability in the given time interval. If the probability exceeds PMAX, return with IRTN=100 without event generation. You need to reduce the time interval DT.
If you set, for example, PMAX=0.1, then you are ignoring the probability (=$\mathtt{PMAX}^2$=0.01) of two (and more) events within DT. A smaller PMAX is safer but more time consuming.

IRR Random number seed.

Output variables

NPH If $\neq 0$, one event is created with NPH laser-photon absorption. If =0, no radiation (following variables are meaningless.)

PE2 Array of dimension (0:3). Final electron 4-momentum (eV/c).

HE2 Final electron helicity. ($-1 \leq$HE2 ≤ 1).

PG Array of dimension (0:3). Final photon 4-momentum (eV/c).

HG Final photon helicity. ($-1 \leq$HG ≤ 1).

PROB Calculated event probability. When an error occurs with IRTN=100, you have to multiply DT at least by a factor PROB/PMAX.

WGT Weight factor of the event. It is always 1 if LENH=0 at the initialization. If not equal to 1, then you should asign a weight $w_0\times$WGT for the final photon end electron and let the initial electron survive with the weight $w_0 \times (1 - \mathtt{WGT})$, where w_0 is the weight of the initial electron before the event. If WGT=1, you should eliminate the initial electron and asign the weight w_0 for the final photon and electron.

IRTN Return code.

Possible errors

IRTN=1000 Initialization not yet done.

IRTN=1001 ξ is larger than XIMAX.

IRTN=1002 λ is larger than LMMAX.

IRTN=100 The total rate exceeds PMAX.

A.1.3 Algorithm

The functions F_{kn} (k=1,2) are stored in a 5-dimensional array FF(k,n,i,j,l) (n=1,MPH), (i=0,MY), (j=0,MXI), (l=0,MLM). The integral over y from 0 to y_i is stored in FINT(k,n,i,j,l). The integral over the full range $0 \leq y \leq 1$ is then FINT(k,n,MY,j,l) For integration, the trapezondal rule is used, which means the function F_{kn} is approximated by a piecewise linear function. The sum of FINT(k,n,MY,j,l) over n is stored in FALL(k,j,l).

For a given initial condition, calculate the parameters ξ^2 and λ and find FALL($*$) by 2-dimensional interpolation. (The asterisk $*$ indicates the appropriate sum over the initial polarization, *i.e.*, $\texttt{FALL}(*)=\texttt{FALL}_{k=1}+h_L h_e \texttt{FALL}_{k=2}$). Then, calculate the total probability P (eq.(138) times the time interval `DT`):

$$P = C_0 \times \texttt{FALL}(*), \qquad C_0 = \frac{\alpha m^2 \xi^2 \Delta t}{\mathcal{E}_{eff}}. \tag{148}$$

Generate a uniform random number r_1 in the interval (0,1). If $r_1 < P$, decide to emit a photon and, otherwise reject.

If rejected, the helicity of the electron should be changed, according to eq.(12), to

$$h_{e,new} = \frac{h_e(1-P_1) - h_L P_2}{1-P}, \qquad P_k = C_0 \times \texttt{FALL}_k \qquad (k=1,2). \tag{149}$$

If accepted, decide how many laser photons to absorb. To do so, sum up $\texttt{FINT}(*,n,\texttt{MY},j,l)$ from $n=1$ to $n=n_1$ until the sum becomes larger than r_1. Then, n_1 will be the number of photons.

Once n_1 is determined, the photon energy is determined by

$$\int_0^y dy\, \texttt{FF}(*) = r_2 \times \texttt{FINT}(*)$$

where r_2 is another uniform random number. The left hand side is known for the mesh point of y (*i.e.*, $\texttt{FINT}(k,n,\texttt{MY},j,l)$). Since we approximate F_{kn} by a piecewise linear function of y, the left hand side is a quadratic function between successive y's. Thus, inverse interpolation with respect to i by solving a quadratic equation gives the photon energy to be emitted.

The helicities of the final photon and electron are calculated from

$$h_\gamma = \frac{h_L F_{3n} + h_e F_{4n}}{F_{1n} + h_L h_e F_{2n}} \tag{150}$$

$$h_{e'} = \frac{h_e(F_{1n}+F_{5n}) + h_L F_{2n}}{F_{1n} + h_L h_e F_{2n}} \tag{151}$$

for $n = n_1$. This is done by directly calling a Bessel function routine.

A.2 Breit-Wheeler Process

A.2.1 Formulas

k,k_L,p,p'	4-momenta of the initial photon, laser photon, final electron and positron.
ω,ω_L,ϵ,ϵ'	Energies of the initial photon, laser photon, final electron and positron.
ξ	Laser intensity parameter
η	Laser energy parameter: $\eta = k{\cdot}k_L/2m^2 \approx \omega_L\omega/m^2$.
n	Number of absorbed laser photons
x	$=p{\cdot}k_L/k{\cdot}k_L \approx \epsilon/\omega$. $(0 < x < 1)$
u	$u = 1/[4x(1-x)]$, $x = \frac{1}{2}[1 \pm \sqrt{1-1/u}]$, $(1 < u < \infty)$.
u_n	Maximum u for given n: $u_n = n\eta/(1+\xi^2)$.

x_n $x_n = \frac{1}{2}[1 - \sqrt{1 - 1/u_n}]$

h_γ Initial photon helicity

h_L Laser helicity

h_e Final electron helicity

$\overline{h}_e$ 'Detector' helicity of the final electron.

θ_e Final electron angle

$$\theta_e \approx \frac{m}{\epsilon}\sqrt{(1+\xi^2)\left(\frac{u_n}{u} - 1\right)} = \frac{m}{\epsilon}\sqrt{4\eta n x(1-x) - (1+\xi^2)}$$

z_n The argument of the Bessel functions in the following expressions:

$$z_n = 2n\frac{\xi}{\sqrt{1+\xi^2}}\sqrt{\frac{u}{u_n}\left(1 - \frac{u}{u_n}\right)}$$

Total number of pair electrons per unit time summed over the positron polarization is

$$W = \frac{\alpha m^2 \xi^2}{2\omega} \sum_{n=1,\ n>(1+\xi^2)/\eta}^{\infty} \int_{x_n}^{1-x_n} dx[G_{1n} + h_L h_\gamma G_{3n} + \overline{h}_e(h_L G_{2n} + h_\gamma G_{4n})] \tag{152}$$

Sum over final electron polarization gives

$$W = \frac{\alpha m^2 \xi^2}{\omega} \sum_{n=1,\ n>(1+\xi^2)/\eta}^{\infty} \int_{x_n}^{1-x_n} dx[G_{1n} + h_L h_\gamma G_{3n}] \tag{153}$$

$$\begin{aligned}
G_{1n} &= \frac{1}{\xi^2}J_n^2 + \frac{1}{2}(2u-1)(J_{n-1}^2 + J_{n+1}^2 - 2J_n^2) && (154)\\
G_{2n} &= (1-2x)2u\left(\frac{1}{2} - \frac{u}{u_n}\right)(J_{n-1}^2 - J_{n+1}^2) && (155)\\
G_{3n} &= -(2u-1)\left(\frac{1}{2} - \frac{u}{u_n}\right)(J_{n-1}^2 - J_{n+1}^2) && (156)\\
G_{4n} &= \frac{1}{x\xi^2}J_n^2 - u(1-2x)(J_{n-1}^2 + J_{n+1}^2 - 2J_n^2) && (157)
\end{aligned}$$

These formulas can be obtained from those of F_{kn} by the replacement

$$\begin{array}{ccccccc}
\omega & \rightarrow & -\omega, & \quad & \mathcal{E} & \rightarrow & -\epsilon \\
h_\gamma & \rightarrow & -h_\gamma, & \quad & h_e & \rightarrow & -h_e.
\end{array} \tag{158}$$

This implies

$$\begin{array}{ccccccc}
v & \rightarrow & -1/(1-x), & \quad & v_n & \rightarrow & -4xu_n \\
v/v_n & \rightarrow & u/u_n, & \quad & z_n & \rightarrow & z_n.
\end{array} \tag{159}$$

For convenience, we have changed the sign as

$$F_{1n} \rightarrow -G_{1n}, \quad F_{2n} \rightarrow +G_{2n}, \quad F_{3n} \rightarrow +G_{3n}, \quad F_{4n} \rightarrow -G_{4n}. \tag{160}$$

For given x and n, the final momenta are given by

$$p = xk + \left[n(1-x) - \frac{4\xi^2}{\eta x} \right] k_L + m\sqrt{4n\eta x(1-x) - (1+\xi^2)}(e_1 \cos\phi + e_2 \sin\phi), \quad (161)$$

$$p' = (1-x)k + \left[nx - \frac{4\xi^2}{\eta(1-x)} \right] k_L - m\sqrt{4n\eta x(1-x) - (1+\xi^2)}(e_1 \cos\phi + e_2 \sin\phi). \quad (162)$$

Here, ϕ is the azimuthal scattering angle in a head-on frame and

$$e_2^\mu = \begin{bmatrix} 0 \\ \frac{\boldsymbol{k}\times\boldsymbol{k}_L}{|\boldsymbol{k}\times\boldsymbol{k}_L|} \end{bmatrix}, \qquad e_1^\mu \equiv -\frac{\epsilon_{\mu\nu\alpha\beta} e_2^\nu k^\alpha k_L^\beta}{k\cdot k_L} = \begin{bmatrix} |\boldsymbol{k}\times\boldsymbol{k}_L| / k\cdot k_L \\ \boldsymbol{e}_2 \times (\omega \boldsymbol{k}_L - \boldsymbol{k}\omega_L)/k\cdot k_L \end{bmatrix}. \quad (163)$$

These vectors satisfy

$$k\cdot e_j = k_L\cdot e_j = 0, \quad e_j\cdot e_j = -1, \quad (j=1,2), \qquad e_1\cdot e_2 = 0. \quad (164)$$

A.2.2 Usage

Initialization

G_{kn} are 3-argument functions. In order to avoid discontinuities due to the n-photon threshold, following variables (in addition to ξ^2) are used as the independent variables instead of (ξ, η, x):

q Defined by $q = \eta/(1+\xi^2)$. The n-photon threshold is given by $q \geq 1/n$.

y Defined by $x = \frac{1}{2} - \frac{1}{2}\sqrt{1 - 1/u_n}y$. $0 \leq y \leq 1$ represents electrons with energy $\leq \omega/2$ and $-1 \leq y \leq 0$ those $\geq \omega/2$. Since G_{1n} and G_{3n} are even functions of y, only the part $y \geq 0$ is tabulated.

Now, the initialization is done by

```
 CALL NLBWST(MY,MPH,MXI,MQ,XIMAX,ETAMAX,LENHBW,ENHBWF,IRTN)
```

MY Number of y's. (i-th y point is $y_i = i/$MY, $i = 0,$ MY)

MPH Maximum number of laser photons. ($n = 0,$ MPH)

MXI Number of ξ's. ($\xi_j^2 =$ XIMAX$^2 \times j/$MXI, $j = 0,$ MXI)

MQ Number of q's. Non-equally-spaced MQ points are selected in $1/$MPH $\leq q \leq$ ETAMAX.

XIMAX Maximum ξ.

ETAMAX Maximum η. Must satisfy ETAMAX $> 1/$MPH. Otherwise, no pair creation is possible.

LENHBW Flag to apply a rate enhancement function.

ENHBWF Enhancement function name declared external. Used when LENHCP$\geq$1.

IRTN Return code.

The parameters MY, MPH, MXI can be different from those for nlcpst. A second call of NLBWST will replace the parameters and the arrays created in the first call.

The enhancement function, if needed, has to be defined as

```
FUNCTION ENHBWF(Y)
REAL*8 ENHBWF,Y
ENHBWF=......
RETURN
END
```

The enhancement function has to be a function of $y = (1-2x)/\sqrt{1-1/u_n}$ $(-1 \leq y \leq 1)$ only and it must be an even function of y and ≥ 1 for all y. (Actually, only the part $0 \leq y \leq 1$ is used.) The probability functions G_{kn} are multiplied by ENHBWF(y). Note that $|y|$ close to 1 represents events with a large unbalance of energy between final electron and positron. See NLBWGN for how the weight should be treated in the simulation.
Possible errors

IRTN=1000 Memory insufficient. You have to reduce MY×MPH×MXI×MQ or increase the parameter MW in the FORTRAN source.

IRTN=1001 to 1004 Either one of MY,MPH,MXI,MQ is too large.

IRTN=1100 The enhancement function less than 1 at some y.

Event generation

An event is generated by

```
NLBWGN(PG,HG,WL,NL,HL,PD,DT,PMAX,IRR,NPH,PELE,HELE,PPOS,HPOS,
          PROB,WGT,IRTN)
```

Input variables

PG Array of dimension (0:3). Initial (high energy) photon 4-momentum (eV/c).

HG Photon helicity. $(-1 \leq$ HG $\leq 1)$.

WL Laser photon energy (eV).

NL Array of dimension 3. Unit vector along the laser direction.

HL Laser helicity (-1 or $+1$).

PD Laser power density (Watt/m^2).

DT Time interval times the velocity of light (meter).

PMAX Maximum probability of pair creation in DT. Same as in NLCPGN.

IRR Random number seed.

Output variables

NPH If $\neq 0$, number of absorbed laser photons. If 0, no pair creation.

PELE Array of dimension (0:3). 4-momentum of final electron.

HELE	Helicity of final electron.
PPOS	Array of dimension (0:3). 4-momentum of final positron.
HPOS	Helicity of final positron.
PROB	Calculated event probability. Same as in NLCPGN.
WGT	Event weight. Same as in NLCPGN.
IRTN	Return code.

The event weight is always 1 if initialized with LENHBW=0. When WGT$\neq 1$ (actually, ≤ 1), each of the final pair should be asigned the weight $w_0 \times$ WGT and the initial photon still be retained with the weight $w_0 \times (1 -$ WGT$)$, where w_0 is the weight of the initial photon before the event. When WGT=1, the initial photon should be eliminated.
Possible errors

IRTN=1000	Initialization nit yet done.
IRTN=1001	ξ is larger than XIMAX.
IRTN=1002	η is larger than ETAMAX.
IRTN=100	The total rate exceeds PMAX.

Useful subroutines

Real∗8 function
NLBWFN(KK,K,NPH,XI,ETA,X)
returns the value of the function $G_{kn}(\xi, \eta, x)$, where k=K, n=NPH, ξ=XI, η=ETA, and x=X=ϵ/ω. The first argument KK selects either the direct calculation using a Bessel function routine (KK=1) or use interpolation of the stored table (KK=2). In the latter case, initialization must be done in advance and K must be 1 or 3.

A.2.3 Algorithm

Because of the threshold behavior, the algorithm is slightly different from that for the Compton process. The mesh for q cannot be equally spaced. Select q's such that all the threshold points $1/n$ ($n = n_{min}, n_{min}+1, \ldots$ MPH, n_{min} = integer part of $1/$ETAMAX) are included, that q are equally spaced between successive thresholds, and that the spaces are not very diffrent. Thus, the total number of q's may not exactly equal to MQ.

The functions stored in the array GG(k,n,i,j,l) are G_{1n} (k=1) and G_{3n} (k=2). The integral GINT(k,n,i,j,l) is calculated as in the Compton case. The sum of GINT(k,n,MY,j,l) over n does not make sense because the result would be a quite discontinuos function of ξ and q, which makes the interpolation inaccurate. However, for given q, the sum from $n = n(q)$ to MPH is continuous where $n(q)$ is the minimum integer which does not exceed $1/q$. Thus, the sum over $n = n$ to MPH is stored in GALL(k,n,j,l).

For given initial condition, calculate ξ and q. Then, interpolate GALL(k,$n(q)$,j,l) for j and l and calculate the total probability P by summing for appropriate polarization:

$$P = C_0 \times \texttt{GALL}(*), \qquad \texttt{GALL}(*) = \texttt{GALL}_{k=1} + \texttt{GALL}_{k=2}, \qquad C_0 = \frac{\alpha m^2 \xi^2 \Delta t}{\omega}. \tag{165}$$

Generate a uniform random number r_1 and reject an event if $r_1 \geq P$. In this case the helicity of the photon should change to

$$h_{\gamma,new} = \frac{h_\gamma(1-P_1) - h_L P_2}{1-P}, \qquad P_k = C_0 \times \mathtt{GALL}_k, \quad (k=1,2). \tag{166}$$

If $r_1 < P$, decide to create a pair. The number of laser photons to absorb is determined from GINT($*$,n,MY,j,l) as in the Compton case.

To determine the electron energy, generate another random number r_2 in the range (-1,$+1$). Then, y $(0 \leq y \leq 1)$ is determined from $|r_2|$ as in the Compton case. Adopt $-y$ instead of y if $r_2 \leq 0$.

The helicity of electron is given by

$$h_e = \frac{h_L G_{2n} + h_\gamma G_{4n}}{G_{1n} + h_L h_\gamma G_{3n}}. \tag{167}$$

The positron momentum is calculated by the momentum conservation and the helicity from the above formula with y replaced by $-y$.

References

[1] K .Yokoya, *A Computer Simulation Code for the Beam-Beam Interaction in Linear Colliders*, KEK report 85-9, Oct. 1985. 3

[2] 3

[3] V. B. Berestetskii, E. M. Lifshitz and L. P. Pitaevskii, *Quantum Electrodynamics*, volume 4 of *Course of theoretical Physics*, second edition translated. Pergamon Press. 45, 58, 58, 58, 58, 76

[4] H. A. Tolhoek, Rev. Mod. Phys. **28** (1956) 277. 58

[5] Y. S. Tsai, SLAC-PUB-5924 Nov. 1992. 77

[6] T. Tauchi, K. Yokoya and P. Chen, Part. Acc. **41** (1993) 29. 25

Index